information

Article

Secure and Efficient Exchange of Threat Information Using Blockchain Technology

Maryam Pahlevan * and Valentin Ionita

Department of Communications and Networking, School of Electrical Engineering, Aalto University, PL 15600, 00076 Aalto, Finland
* Correspondence: maryam.pahlevan@aalto.fi

Abstract: In recent years, sharing threat information has been one of the most suggested solutions for combating the ever-increasing number of cyberattacks, which stem from the system-wide adoption of Information and Communication Technology (ICT) and consequently endangers the digital and physical assets of organizations. Several solutions, however, were proposed to facilitate data exchange between different systems, but none were able to address the main challenges of threat sharing such as trust, privacy, interoperability, and automation in a single solution. To address these issues, this paper presents a secure and efficient threat information sharing system that leverages Trusted Automated Exchange of Intelligence Information (TAXII^{TM}) standard and private blockchain technology to automate the threat sharing procedure while offering privacy, data integrity, and interoperability. The extensive evaluation of the solution implementation indicates its capability to offer secure communication between participants without sacrificing data privacy and overall performance as opposed to existing solutions.

Keywords: threat sharing platform; blockchain; TAXII standard

Citation: Pahlevan, M.; Ionita, V. Secure and Efficient Exchange of Threat Information Using Blockchain Technology. *Information* **2022**, *13*, 463. https://doi.org/10.3390/info13100463

Academic Editor: Panagiotis Trakadas

Received: 9 August 2022
Accepted: 22 September 2022
Published: 28 September 2022

1. Introduction

In the modern era, critical infrastructures comprise a set of cyber-physical systems which play key roles in maintaining healthy and functional societies [1,2]. While the rapid adoption of Information and Communication Technology (ICT) systems by critical infrastructures such as power networks have enhanced the performance and efficiency of the overall systems, it has also opened the door to a novel type of cyberattacks that could cause irreparable physical damages [3].

Inspired by the hacker's strategy where they form well-structured groups and build a comprehensive knowledge base for their most effective practices, several studies proposed cyber threat information sharing as one of the most promising solutions to eliminate complex modern cyberattacks [4,5]. According to NIST [6], Cyber Threat Information (CTI) is "any information that can help an organization identify, assess, monitor, and respond to cyber threats". In other words, CTI creates an opportunity for detection and response to incidents in a timely and actionable manner. Thereby, sharing CTI enables organizations to achieve robust cybersecurity defence, effective recovery capacity, and uninterrupted operation flow [4].

Like any information sharing procedure, there are essential requirements for sharing CTI between organisations and security experts. Amongst these needs, establishing trust is seen as a key enabler in such relationships. Consequently, in recent years, blockchain has been widely deployed in modern information sharing systems offering trust between diverse participants in the absence of a trusted central authority [7]. Under this perspective, several works proposed blockchain-based CTI sharing platforms to address trust issues in such systems. Nevertheless, the developed solutions trade performance and privacy for secure communication since trusted relationships, transparency, and accountability in blockchain come at expense of performance and the lack of privacy [8].

This work builds on our previous solution [9] in which the technological capacity of the TAXII framework was expanded to address the inherited deficiencies of the platform such as the lack of support for event-driven architecture. The effectiveness of the CTI sharing platform in addition to trust formation between different parties highly relies on the reliability of shared information. Thereby, the developed solution leverages the TAXII framework, blockchain and smart contracts to address the following research questions:

- How does the threat information sharing platform support secure communication between diverse participants?
- What are the trade-offs between trust, immutability and privacy in the process of sharing CTI and how can these needs be addressed effectively in a single solution?
- What is the overhead the involved parties would incur due to the deployment of the developed solution?

The contributions of this paper are:

- The developed solution addresses trust issues associated with the CTI exchange on two levels: (1) the dependability of a publisher; (2) the reliability of shared information.
- The solution records the hash of threat information immutably instead of the actual CTI, thus it complies with the General Data Protection Regulation (GDPR) [10] right to be forgotten.
- An implementation is entirely built on open-source software including the official TAXII framework, Hyperledger Fabric and Quorum.
- The solution is thoroughly assessed in two different ways: A quantitative evaluation using indicators such as resource utilisation and latency and a qualitative evaluation considering attack vectors.

The rest of the paper is structured as follows: Section 2 briefly discusses the concepts used throughout the paper. The blueprint of the developed solution is described thoroughly in Section 3. Section 4 presents the implementation details of the design which are laid out in the previous section. The extensive evaluation of the prototype is presented in Section 5.

2. Background

2.1. Blockchain Technology

Blockchain is defined as a distributed database where different nodes collaborate in the absence of a trusted third party. To establish trust among unknown nodes, blockchain deploys a consensus algorithm, enabling a legitimate node to add a new block in the correct order. Data blocks are immutable in blockchain compared to standard Distributed Data-Base (DDB). More precisely, users in blockchain can only execute write and read operations while Distributed Data-Base Management System (DDBMS) apart from writing and reading allows carrying out an update and delete operations on data records. To support this feature, each data block in the blockchain comprises a timestamp, a cryptographic hash of the current block as well as the parent block and transaction information [11]. Thereby, any changes in the content of the block cause a mismatch between the block's hash and the value stored in the successor blocks. Additionally, in blockchain, only the owner of data can modify the information relating to the ownership while DDBMS allows the system administrator to change the ownership of any record. Thus, in blockchain the provenance of a record is traceable. Similarly to DDB, blockchain offers availability and security resilience against arbitrary failures since all nodes in the blockchain keep identical data blocks. Along with security enhancement, blockchain utilises cryptographic primitives to create user identifiers in form of addresses and thus improves the user's privacy significantly [12]. Furthermore, many blockchains allow distributed applications, also called smart contracts, to run on their nodes. Smart contracts are mainly utilised for identity and digital signature verification, computation tasks, and interactions with other smart contracts. All transactions triggered by a smart contract, are executed by all (full) nodes in the blockchain in a reliable and deterministic manner. Additionally, all these transactions are persisted on the blockchain and can be referred later for dispute management. It must be noted that even

though the content of the smart contract is deployed immutably by all nodes, the outcomes of transaction execution are only recorded on a chosen node.

In the last decade, a wide range of blockchains has been developed where each serves disparate needs. For instance, blockchains based on their access control management strategies are categorized into permissioned, permissionless, private and public blockchains. A private permissioned blockchain only allows authorized nodes to execute transactions and read data blocks. Thereby, it employs a lightweight consensus algorithm to establish trust between different nodes. On the other hand, there are no restrictions on joining and subsequently performing different transactions in a public permissionless blockchain. Therefore, it deploys a compute-intensive consensus mechanism to address trust issues, leading to prohibitively expensive execution of transactions [13].

With what is described above, blockchain is deemed as an appropriate solution for sharing critical information where trust, integrity, availability, and traceability are indispensable attributes.

2.2. Cyber Threat Information Sharing

This section explains the related work concerning Cyber Threat Information Sharing methods. It also lists the criteria which are used to evaluate CTI sharing frameworks.

As stated in the 2019 cyber-crime study conducted by Accenture [14], there is a substantial growth in the financial loss that organisations incur due to different cyberattacks. The study also predicted that in the next five years cyber threats jeopardize the assets of companies exposed to this class of attacks with an estimated value of $5.2 trillion. Thereby, the numbers mentioned above highlight the definite need for robust and reliable procedures aimed at securing organisations and critical infrastructure against complex cyberattacks. Standard information security mechanisms and technologies such as firewalls have been widely deployed in critical infrastructures as mitigation for malicious activities, but they are, nevertheless, inadequate to counter correlated and sophisticated cyberattacks [5].

A vast majority of organisations collectively agreed that sharing threat information is one of the most effective solutions to combat modern adversaries [15]. This statement had been backed by the SANS study [16], wherein about 81% of security experts mentioned that they leverage shared CTI to achieve better cyber-security resilience. Furthermore, in the same study SANS demonstrated that over the last few years the amount of produced and consumed threat information has been increasing continuously. This trend implies a massive growth in the number of organisations that are actively contributing to sharing information.

Establishing a successful CTI sharing system relies on several factors. The respective requirements and challenges are extensively discussed in [16–19]. These studies viewed threat sharing as community-driven efforts which, to be effective, entail active contributions between public and private organisations. To achieve this, sector-specific Information Sharing and Analysis Centers (ISACs) such as European Energy—Information Sharing Analysis Centre (EE-ISAC) have been established with the goal of interconnecting different governmental and industry partners [5].

In [6], NIST also defined a set of guidelines for establishing an effective CTI sharing framework. Additionally, it attributed the reluctance of organisations to share threat information to the following factors: trust issues, privacy concerns particularly regarding classified and personal data, lack of automated and interoperable sharing mechanisms, the asymmetric relationship between CTI publisher and consumer, reliability of shared information, regulation, and traceability issues. Among the mentioned obstacles, interoperability, trusted relationships, privacy issues, the integrity of information and accountability have been drawing significant attention and thus have been addressed in many research works [20–22].

Despite an ever-increasing volume of shared CTI amongst organisations over the last decades, still, a significant percentage of the information is unstructured [23]. To address the interoperability issues, in recent years many data models such as Open Incident of

Compromise (OpenIOC) [24], Cyber Observable eXpression (CybOX^{TM}) [25], and Structured Threat Information eXpression (STIX) [26] have been developed and have presented indicators of compromise, attack patterns and countermeasures. The majority of the CTI ontologies, however, were tailored to a specific use case, STIX data model [26] were designed to cover a full range of topics within the cyber-security realm. Furthermore, the TAXII standard [27] was defined to support the secure exchange of threat information particularly data structured in STIX formats. Our previous work [9], which the current implementation is also built upon, expanded the official TAXII framework, which is encouraged to be utilised by public and private sectors for sharing threat information.

2.2.1. Blockchain-Based Cyber Threat Information Sharing

A substantial percentage of publications on blockchains, specifically Bitcoin, have been predominantly centred around cyber-security topics [28] since blockchain, as described in Section 2.1, offers security properties such as trust, traceability, accountability, and reliability in a single solution. From the topics discussed in the studies, blockchain-based cyber security mechanisms that are utilised for information sharing and auditing, after security methods aimed at IoT devices, represent the most dominant theme in this field [29]. For instance, authors in [30] developed a novel CTI sharing platform using the STIX 2.0 standard and Hyperledger Fabric [31], tackling critical barriers in sharing information namely interoperability issues, trust and privacy concerns. This solution leveraged the Fabric channels and smart contracts to achieve higher efficiency and better efficacy while exchanging CTI between different parties. In [13], a private permissioned blockchain is leveraged to share threat information between users and service providers.

In [32], authors introduced a new blockchain-enabled method for sharing incidents concerning cloud services. Hajizadeh et al. [33] devised an efficient and secure CTI sharing system with the use of Hyperledger Fabric and Software-Defined Networking (SDN) [34]. Thereby, this work benefited from the security properties of a private blockchain and the management capacities of SDN in a single solution, guaranteeing the integrity of threat information and shortening the time between detection and mitigation activities. Furthermore, the implementation has been thoroughly evaluated using the performance indicators such as throughput and latency of transactions aimed at writing and reading data from the fabric channels. All aforementioned solutions stored the full CTI record in the blockchain, which made them less scalable considering the ever-increasing volume of cyber incidents and low speed of blockchain transactions. Additionally, they undermined the GDPR right to be forgotten [35] by keeping the immutable logs of CTI records.

In [36], a distributed threat information sharing framework based on blockchain is proposed. This work devised two different smart contracts to enable a fully supervised decision-making process and access control management. Wu et al. [37] studied trust issues about threat sharing from two distinct perspectives: trustworthiness of participants and quality of information. To this end, they implemented a quality assessment platform using Ethereum and Trusted Execution Environment (TEE) where each participant runs the reputation algorithm independently on their TEE and later shares the outcome with others via Ethereum. Both works described above lacked empirical deployments and subsequently did not present the proper evaluations for the developed solutions.

As discussed in [38], data gathering plays a key role in threat-sharing platforms, particularly in modern systems where the ever-increasing volume of collected data has been stored in external storage such as cloud servers. Cha et al. [38] addressed the reliability issues arising from storing threat information in cloud servers with the use of private blockchains. To this end, the solution recorded the essential data required for integrity checks such as hash in a blockchain while storing the actual data in cloud servers. Thereby, it remarkably improves the efficiency of the overall system by leveraging cloud servers and private blockchain simultaneously. Despite the mentioned advantages, this work did not discuss the negative impacts that the overhead stemming from the interaction with a blockchain would have on the CTI sharing process.

To address deficiencies of state-of-art solutions, we developed a blockchain-based secure and efficient threat-sharing platform that leverages the security properties of the blockchain by recording hash objects in the blockchain while optimizing the overall performance by storing data in a database. Furthermore, our design has been evaluated extensively from different dimensions including performance and cyber resilience.

2.2.2. Evaluation Criteria for Cyber Threat Information Sharing

Coupled with the essential requirements of CTI sharing platforms that were described in Section 2.2, the evaluation criteria for such systems can be listed as follows:

1. Integrity: the authenticity of CTI is preserved while exchanging amongst organisations.
2. Performance: a CTI sharing platform optimises the performance merits such as throughput and latency.
3. Trust: only authorised organisations can participate in the threat-sharing process.
4. Privacy: the platform does not expose any personal data to unauthorised parties.
5. Automation: Upon publishing new threat information, all organisations that are subscribed to the respective incident should automatically receive the information.

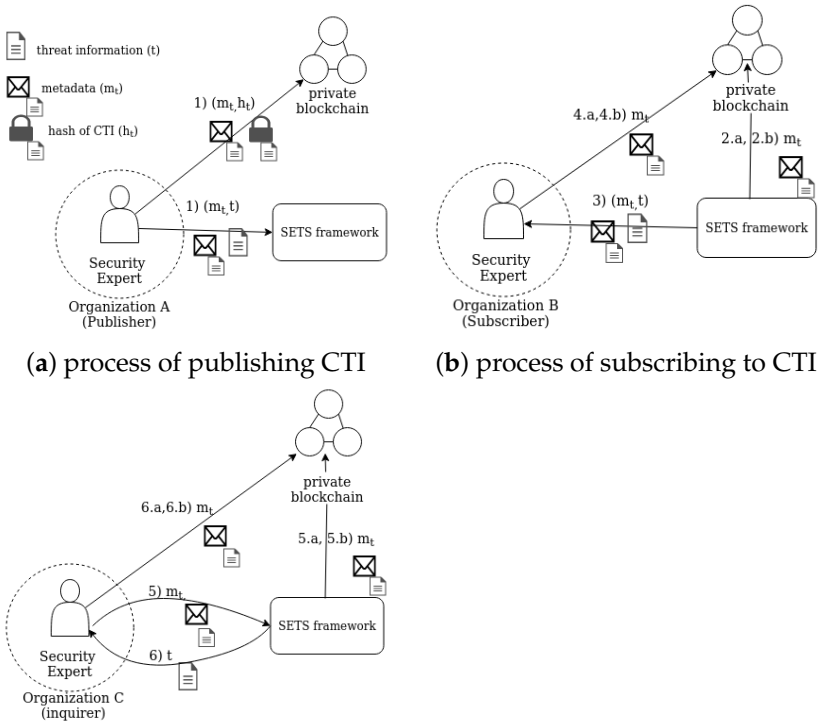
3. Secure and Efficient Threat Information Sharing

Considering the growing concern about data security and privacy, this work develops a blockchain-enabled Secure and Efficient Threat Sharing platform (SETS) which takes into account the data protection regulations, specifically the GDPR right to be forgotten, while leveraging a private permissioned blockchain for security properties. To this end, it extends our previous solution [9] wherein the technological capacity of the TAXII framework was expanded to address the inherited deficiencies of the platform including lack of support for the event-driven architecture and absence of database protection. The rationale behind the framework selection was to address interoperability issues among different sectors since the TAXII framework is deemed as one of the most prominent cybersecurity information sharing platforms.

It is noteworthy that the effectiveness of the CTI sharing platform not only depends upon forming a trusting relationship between participants but also relies highly on the reliability of shared information. Therefore, a security professional, before acting upon received information, first would assure the trustworthiness of a CTI publisher, and then would ensure that the integrity of the information has been preserved throughout the sharing process. To enact the described principles, our solution only allows authenticated and authorised producers to publish CTI in the SETS framework. Additionally, a publisher stores the hash of information in the blockchain which would be later utilised by a consumer to validate the integrity of a received CTI. Thereby, the developed solution benefits from the tamper-proof logs, transparency, and traceability features of the blockchain while minimizing the performance loss spawning from the interaction with the blockchain, by recording only the essential data on the blockchain.

The Design

This section explains the design of the SETS framework that was briefly introduced in the previous section. All steps in the developed solution, from publishing threat information to distributing it to the interested organisations, fulfil the criteria mentioned in Section 2.2.2. These steps are summarized in Figure 1.



(**a**) process of publishing CTI (**b**) process of subscribing to CTI

(**c**) process of inquiry CTI

Figure 1. High-level view of the SETS framework. This figure illustrates how different components of the framework communicate with each other in the context of publishing, subscribing and inquiry processes.

The SETS framework only accepts connections from organisations that possess valid credentials. Furthermore, it grants permission for publishing or inquiry data based on the organisation's role and thus satisfies the third criterion which is trust.

In the developed solution, a producer persists the hash of CTI in a private blockchain while publishing the actual threat information in the SETS framework to meet the second criteria which is performance, and the fourth criterion is privacy. Consequently, on the receiving side, the organisation verifies the integrity of CTI using the respective hash stored in the blockchain which leads to meeting the first criterion which is integrity.

In [9], the TAXII framework was extended with the publish-subscribe module, enabling the automatic distribution of CTI towards the subscribed partners upon reception of new threat information. Thereby, the SETS framework satisfies the fifth criterion which is automation.

The process of sharing threat information in this paper which is also illustrated in Figure 1 can be described as follows:

0 **Discovery**: A Security Expert (SE) in Organisation A discovers a cyber incident. t and m_t represent the threat information and the related metadata, respectively. m_t, which comprises the CTI identifier, collection ID and version of the object, is mainly used to identify the CTI object uniquely. According to [39], a CTI identifier is generated in form of object-type–UUIDv4 where object-type specifies the value of a certain type of CTI object and the UUIDv4 is generated based on RFC 4122 [40]. The collection ID also determines to which collection the information belongs. SE computes the hash of the information, h_t, using the hash function $H()$. In the next steps, the hash value would be utilised for the integrity check.

1 **Publish**: SE publishes the new threat information t to the SETS framework where t is stored in the database to satisfy the performance criterion. On the other hand, the expert writes m_t along with the h_t (the tuple (m_t, h_t)) in the private blockchain where only authorised partners can execute transactions (i.e., the trust criteria). Despite sev-

eral advantages of storing threat information in the private blockchain, this approach has some shortfalls such as lower throughput and immutable records compared to the database. Thereby, this solution leverages the database and the blockchain simultaneously to achieve better performance while protecting the data integrity against unsolicited malicious activities. Furthermore, the rationale behind choosing the private blockchain over the public one is to avoid the high cost of transaction executions in terms of fee and time in the public blockchain.

2.a **Admission**: Upon reception of a CTI object, the SETS framework first ensures that the information has not been published before. Then it retrieves the respective hash h_t from the private blockchain using the information's metadata m_t. Next, it computes the hash of CTI with the use of $H()$ and checks $H(t)$ against the hash value from the blockchain. The SETS framework stores the tuple (m_t, t) in the database after the successful integrity check i.e., if $H(t) == h_t$.

2.b **Rejection**: if the CTI object either has been received before or the hash values are mismatched (i.e., $H(t) \neq h_t$), the TAXII framework rejects the CTI submission from SE. However, it does not take any further action to notify SE about the incident. In the latter case, the hash mismatch implies that the CTI object has been altered while being sent to the SETS framework.

3 **Distribution**: Apart from the integrity check and persisting t in the database, the SETS framework also distributes the threat information to the interested organisations via the pub-sub module. Since response time is crucial in effectively mitigating cyberattacks, the CTI object is forwarded to the pub-sub module immediately after receiving the SE's submission. Therefore, the SETS framework does not wait for the completion of the integrity check which might take a longer time due to interaction with the blockchain. Therefore, at the time of distribution, it is quite likely that the authenticity of t is not validated yet. Additionally, the content of the CTI object in transition is potentially the attackers' alteration target. Thereby, the organisation on the receiving end needs to check the integrity of the CTI object upon reception.

4 **Subscribe**: A security expert in Organisation B subscribes to a certain type of threat information (e.g., an indicator of compromise) which is shared with the SETS framework by different organisations. Then it will receive the relevant CTI from the SETS framework when any threat information is published in the respective channel. Similarly to the SETS framework, SE computes the hash of CTI using $H()$ and compares it with the hash value that is retrieved from the blockchain. If $H(t) == h_t$, SE accepts the CTI object and takes it into use for further actions (step 4.a), otherwise it rejects the SETS' submission (step 4.b) since the hash mismatch implies that the CTI object has been altered while transporting.

5 **Inquiry**: In addition to the publish-subscribe communication paradigm, the SETS framework provides a specific CTI object in response to the SE's request. In this communication model, the SETS framework retrieves the queried threat information t from the database. Next, it computes the hash of CTI and checks it against h_t which is obtained from the blockchain. Like the distribution phase, the SETS framework does not wait for the completion of the integrity check and sends back t immediately after retrieval from the database. The SETS framework marks the CTI record as valid if $H(t) == h_t$ (step 5.a), otherwise the record is marked as invalid (step 5.b), implying the CTI object in the database has been altered. Therefore, the SETS framework will reject future queries about the same CTI record.

6 **Inquiry Response**: SE upon reception of a response to the queried CTI, retrieves the hash of data from the blockchain and checks it against the hash value that is computed using $H()$. Like the subscribe phase, if $H(t)$ matches h_t, SE accepts the CTI object (step 6.a), otherwise, it rejects the SETS' reply (step 6.b), implying the CTI object has been modified either in the database or in transition.

4. The Implementation

This section details the prototype implementing the design described above. This prototype is built on top of the official implementation of TAXII 2.1 standard which is publicly available at OASIS TC open repository [41,42], Hyperledger Fabric and Quorum as private permissioned blockchains. As illustrated in Figure 2, the prototype is composed of four key components: TAXII client, TAXII server (i.e., Medallion node), pub-sub middleware and private blockchain.

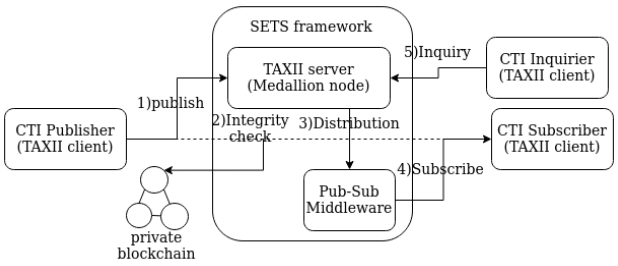


Figure 2. Architecture of the SETS framework. The figure depicts the key components of the prototype. The step numbers match Figure 1.

4.1. TAXII Client

Each organisation with the use of diverse information security tools produces a huge volume of CTI such as anomaly reports and subsequently shares the data with other organisations and legislative bodies that could benefit from this exchange. To enable information sharing between diverse parties, every organisation, regardless of the roles they may play in the context of CTI exchange, must support the TAXII client. In this prototype, the client component extends a reference implementation of TAXII 2.1 client [42] with the necessary functionalities that facilitate communication with blockchain and pub-sub middleware. The TAXII client may take on different roles, namely producer and consumer. If the client produces a CTI object, it disseminates the threat information to the TAXII server which is mainly responsible for retaining CTI records and relaying them to the subscribed parties. Additionally, to guarantee the integrity of the data in transition, the client computes the hash of the CTI object using the hash function and further writes the hash value along with the object's metadata in a dedicated blockchain. To perform the latter step, the blockchain-specific adapters for Quorum [43] and Hyperledger Fabric [31] are added to the standard implementation of the TAXII client as these blockchains are among the most widely deployed private permissioned blockchains. These adapters provide interfaces to the `data-storage` smart contract that is deployed in the respective blockchain. The `data-storage` smart contract is composed of the following functions:

- `WriteCTIHash`: this function stores the tuple m_t, h_t, which is given as input in the dedicated blockchain.
- `ReadCTIHash`: this function retrieves the hash value h_t, which is mapped to the given metadata m_t from the blockchain.

Therefore, these adapters enable running transactions on either Quorum or Fabric nodes by invoking the functions described above. It is worth noting that the client only persists the hash of a CTI object in the blockchain via `WriteCTIHash`, since the threat information might directly or indirectly contain some personal data and thus recording it on the blockchain would violate the GDPR right to be forgotten.

The TAXII client, apart from the producer role, might subscribe to a certain channel provided by the pub-sub middleware. More precisely, the client creates temporary queues for each CTI class (e.g., attacks) they subscribe to, binds them to the respective channels and begins consuming the information forwarded to these queues.

Upon receiving a new CTI object in a queue, the TAXII client invokes the `ReadCTIHash` function from the smart contract via the corresponding blockchain adapter. Next, it calculates the hash of the object using the hash function and compares it with the hash value

that is obtained from the blockchain. If the authenticity of the CTI object is verified, the client accepts the information or otherwise discards it. Figure 3 illustrates all steps that would be taken within the context of the publish-subscribe communication model by the TAXII client and server.

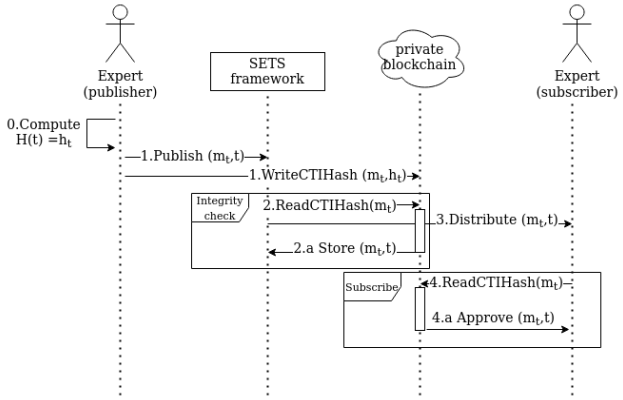


Figure 3. Sequence diagram of the SETS framework concerning the publish-subscribe communication paradigm. Each step in this communication model is shown using the given inputs and the step numbers specified in Figure 1.

4.2. TAXII Server

As stated above, the TAXII server sits between CTI providers and consumers and maintains a repository of CTI objects which would be on-demand forwarded to the interested parties. The server component extends our previous work [9] which addressed the inherited deficiencies of the minimal implementation of TAXII 2.1 server [41], so-called medallion node, with the blockchain integration and pub-sub middleware. Thereby, the TAXII server component similarly to the official implementation gives the ability to clients to access the endpoints defined in TAXII 2.1 protocol using the Flask web framework [44]. It is noteworthy the TAXII server implementation opted for Flask as a web server due to its lightweight design, minimal dependencies on external libraries and modularity. Furthermore, the TAXII server implements different back-end "plugins" where CTI objects and respective metadata are stored. Consequently, a client writes or reads threat information from the backend plugins via disparate interfaces called TAXII Collections. Like our previous work, the server utilises the OAuth2 service Keycloak [45] for the authentication and authorization of clients. Additionally, the server deploys the MongoDB backend for persisting CTI objects.

The TAXII server, upon reception of new information, runs the following tasks in parallel, aiming to minimize the performance loss emerging from the interaction with the dedicated blockchain:

- **Publish**: the server passes the threat information to the pub-sub middleware where CTI objects would be written to pre-defined channels. The server component uses the pub-sub middleware which was implemented in [9] on basis of the RabbitMQ messaging platform.
- **Persist**: the server stores the CTI object in the backend plugin. However, it is most likely that the validity of the record has not confirmed when the `Persist` task was finalized. Therefore, the status of the CTI record is marked as **pending** before receiving the integrity check outcomes.
- **Integrity check**: the server retrieves the hash of the CTI object from the blockchain. To this end, like the client component, the blockchain-specific adapters are added to the server implementation and used to invoke the `ReadCTIHash` function from the `DataStorage` contract. The server also computes the hash of the object and checks it against the hash value acquired from the blockchain. If the hash values match, the status of the corresponding record sets to **finalized**. On contrary, for mismatching

values, the status is set to **invalid**. The integrity check is typically completed after the `Publish` task, thus the CTI consumers always check the authenticity of the received information independently.

The TAXII server, in addition, to publishing requests, might receive an inquiry about a specific CTI object. In such cases, the server executes the integrity check in parallel with the data retrieval from the backend. More precisely, the server retrieves the queried CTI object from the backend and subsequently sends back the information to the client. Simultaneously, the integrity check process that is identical to what was described above is executed and the record's status would be updated accordingly. Figure 4 depicts the steps that would be executed during a CTI inquiry process.

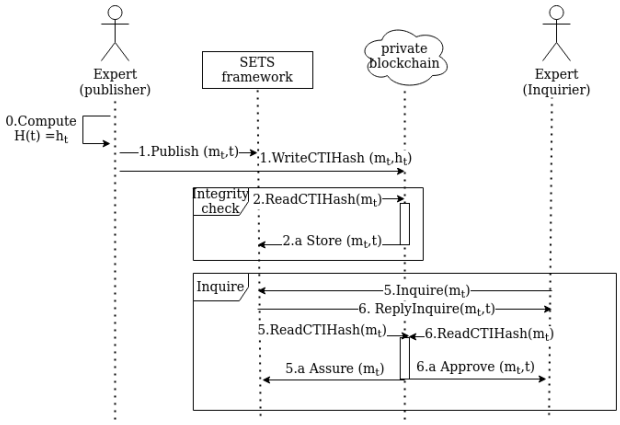


Figure 4. Sequence diagram of the SETS framework with respect to the request-response communication paradigm. Each step in this communication model is shown using the given inputs and the step numbers specified in Figure 1.

4.3. Blockchain-Specific Interface

As described in Sections 4.1 and 4.2 and shown in Figure 5, the blockchain-specific adapters enable communication between the components within the SETS framework and blockchain nodes. Thereby, these adapters define interfaces toward different types of blockchain. In this prototype, the SETS framework only provides the adapters for Quorum and Hyperledger Fabric, although support for other blockchains can be seamlessly incorporated.

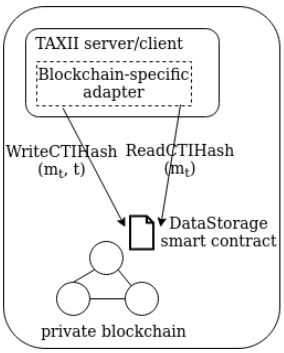


Figure 5. The high-level view of a blockchain-specific adapter.

During the initialisation phase, each adapter establishes a connection between the TAXII components and the respective blockchain. Afterwards, the adapter, based on the operations requested by the component, would trigger varying transactions on the corresponding blockchain. To launch a transaction, the adapter invokes one of the functions that are defined in the `DataStorage` smart contract.

5. Evaluation

5.1. Testing Environment and Setup

The system under test (SUT) as shown in Figure 6 is a genuine representation of the previously presented SETS framework. It includes a TAXII server (with a Pub-Sub middleware and backend plugin), a private blockchain and a pair of experts which play the roles of a producer and a consumer. This way, we can control all parts of the system and measure the impact of the blockchain on the SUT performance. The load for each test represents 1000 messages sent by the publisher and registered by the consumer at various input send rates.

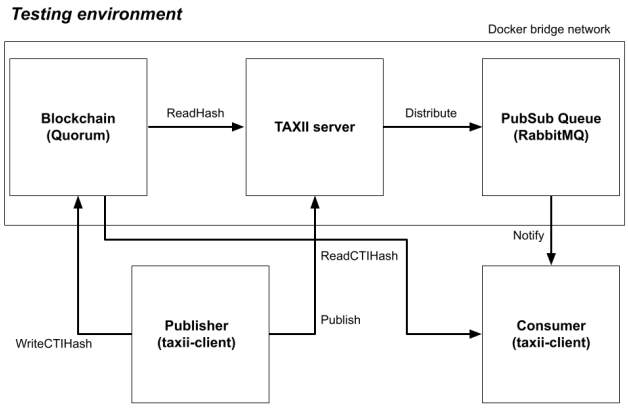


Figure 6. SUT environment.

The evaluation has been performed on a single consumer-level machine, with Ubuntu 20.04.4 LTS OS and an i7-6700HQ, 3.5 GHz, 4 cores, and 8 threads processor. The machine had a score of 954 single-core and 3394 multi-cores when benchmarking using Geekbench 5. The available memory is 16 GB RAM.

The solution has been deployed as several Docker services running on a single host. In the Quorum network, 4 nodes, each with their transaction manager, handle the private smart contract transactions. The consensus algorithm used was Istanbul Byzantine Fault Tolerant (IBFT), using a block time of 1 s, which is the default value for the IBFT mechanism. RabbitMQ was deployed as a Pub-Sub middleware between clients. During testing, the TAXII server has been using the in-memory backend plugin to record messages. Finally, we have run prototypes for the Expert parties (publishers and subscribers), starting from the `taxii2-client` Python package. While the Quorum and the SETS framework are part of the same bridge network, the clients have been running on the host network. Using this network configuration, we are imitating a real-life scenario in which a client would address the parties by specific IP-port addresses, not translated hosts, as it usually works in Docker bridge networks. According to [46], using Docker containers for real-time software development would add under 150 µs RTT (Round-Trip Time), compared to a bare-machine deployment, therefore, the impact on performance is far outweighed by the gained ease of development.

Resource monitoring has been executed by collecting statistics offered by the Docker daemon, using Google's cAdvisor, in a custom setup specifically built for host monitoring. We have recorded the minimum, maximum and average values for CPU and memory usage for the relevant containers.

5.2. Latency, Throughput and Resource Utilisation in SETS framework

An event is a piece of cyber threat information that is meant to be shared using the described system, encoded as STIX objects. To be able to send events to the SETS framework at different rates, the producer uses the `threading` Python module and created a separate thread for each event created. This is like a real environment in which an automated

system would gather the information for an event on the main thread while sending them on a secondary thread. We can introduce an artificial delay between the messages by increasing the time interval between two threads sending their messages, but that does not guarantee the order or resulting send rate, even though it influences the latter. The consumer processes queued messages on a separate thread using the Python RabbitMQ adapter `pika` [47].

Running test cases has been automated using a script where the number of producers, consumers, inter-message interval, and the number of messages, have been given as inputs. The consumer will automatically shut down if it records the expected number of events successfully. The whole test will be shut down forcefully by test scaffolding after some predetermined time. For the initial runtime limit, we have used random values for each set of specific parameters, allowing the system to finish the job, and to have priors for the following tests; thus, these initial tests are not included in the measurements, as they might have had early cutoffs.

For the following results, we have run a setup with 1 producer, 1 consumer and 1000 events. We have used various delay intervals to get as possible to the desired send rates of 50, 75, 100 and 125 messages per second. In 2022, the current year at the time of writing, it is estimated in [48] that about 44 records are exposed every second to hacking attempts. We have chosen the specific send rates in our tests to focus on the above threat level and to be able to future-proof our system by showing it can function just as well with higher event send rates. Therefore, we have started with 50 events per second as the lowest publishing rate for testing, going as high as 125 events per second. We also had in mind similar studies [49] for input send rate comparison, where they use 25 and 50 events per second as the lowest rates when evaluating blockchain performance. Moreover, it's likely that not every event is to be reported individually by a security expert, as some of them might refer to the same attack or vulnerability, with the result of the actual send rate to be supported by the SETS framework being lower than the one reported above. Another aspect is that, in a real-life system, the events would be spread out in many categories, while in our test cases all events reported were put in a single queue in the Pub-Sub middleware. This has the potential of both increasing the consumption rate (thus lowering the latency) because the subscription model receives messages sequentially, and also increasing the success rate, as we have discovered that, even if all the publishing is executed correctly for all events, some get lost while notifying the consumer.

You can see in the tables below (Tables 1 and 2) the throughput and latency values for the achieved send rates. Each send rate is the average of 3 repetitions with the closest rate. Moreover, the first half of the tests that stand as a baseline for performance and resource utilisation comparisons, have been carried out for the SETS platform without a blockchain deployment, and thus, without our added functionality. The latency (Figure 7) is measured as the difference between the last event completely consumed (including verification when using blockchain) and the first event published, over the number of successful events. The throughput (Figure 8) is the number of successfully consumed events over the total consumption time. When comparing our latency measurement results with [49], in which they have a similar, but not identical setup, using IBFT consensus in a 4-node Quorum network, they reach 1.7 to 2.5 s latency per transaction for similar send rates. While they are only measuring the transaction time covering the blockchain only, we cover a much more complex workflow, but we get similar results: 3.05 s latency in our case when the SUT uses the blockchain and under 0.12 s latency when it is not. We can conclude that the added 2.92 s is added by the use of Quorum and the interaction time of all other components (when writing the hashes and reading them for verification) with it.

Table 1. Latency and throughput measurements without blockchain.

Send Rate (m/s)	Success Rate	Throughput (m/s)	Latency (s)
49.06	0.99	9.09	0.10
76.13	0.98	8.94	0.11
102.46	0.97	9.01	0.11
120.96	0.97	8.89	0.11

Table 2. Latency and throughput measurements with blockchain.

Send Rate (m/s)	Success Rate	Throughput (m/s)	Latency (s)
49.21	0.99	0.33	3.00
72.69	0.98	0.32	3.03
96.15	0.96	0.32	3.09
120.04	0.96	0.32	3.09

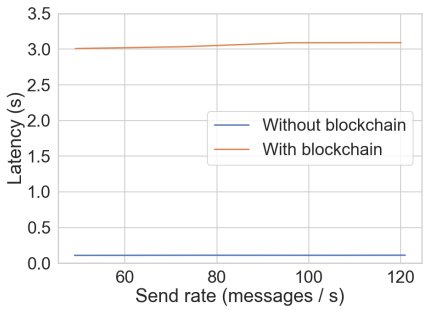


Figure 7. Latency comparison depending on the send rate.

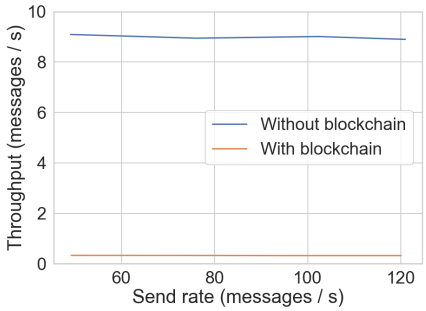


Figure 8. Throughput comparison depending on the send rate.

For the baseline configuration, the average throughput is 8.98 ± 0.12 messages per second (Figure 8) and 0.11 ± 0.00 seconds latency (Figure 7). This is irrespective of the send rate, showing that there is an upper-performance limit which is quite low for the preliminary configuration of the SETS framework. As expected, when we add new security-improving functionality based on blockchain, that performance drops significantly by 2.94 s for latency. The achieved performance for the blockchain tests is 0.32 ± 0.00 messages per second throughput (Figure 8) and 3.05 ± 0.04 seconds latency (Figure 7). While the latency increase and throughput degradation were to be expected, we can say that the system continues to perform consistently, underlining that the added functionality is stable. The success rate (the fraction of the messages successfully recorded by the consumer from all the messages sent by the producer) in all tests is always similar and above 0.95. However, we can observe a slightly decreasing trend with the increase in send rate, but with low effects. The difference between values of success rate (i.e., on average 0.95) and throughput (i.e., on average 0.32 messages per second) is stemmed from the fact the success rate represents the fraction of the messages that are successfully processed by the consumer during the

test execution time while throughput presents the number of messages that are delivered to the consumer in every second. Thereby, the experimental results indicate that for high send rates (e.g., 50 messages per second) the prototype is incapable of processing messages at the same rate as it receives them due to the usage of a slow web framework (i.e., Flask) and several interactions with the Quorum network. Consequently, the SETS framework implementation to be fit for deployment in production environments must undergo a series of improvements.

Following, we have the aggregated resource utilization statistics for the same set of tests. Both the CPU and memory values are computed in the following way: for each test, we have summed the maximum resource utilisation values of the SUT services (producer, consumer, server) and we have averaged those sums for each target send rate. We have considered only these components because, while the Quorum network is an important aspect of the system, it will be, in a real-life scenario, deployed as several independent nodes, each handled by a different trusted party, such that none of them will have to support the full computational effort of the network, while, most likely, each party would act as both a publisher and a subscriber in different scenarios so they would run clients for both. Moreover, the server itself is an integral part of the SETS system and we are looking to assess its resource usage when introducing the blockchain.

From the results above, we can see a clear difference between the resource utilisation of the system without blockchain (Table 3) and with (Table 4). Running an independent t-test on the original test values (before averaging), we obtain a t-statistic of -18.57 (p-value of 6.23×10^{-15}) for the CPU and of 1.77 (p-value of 0.08) for the memory usage. Therefore, we can conclude that the CPU usage (Figure 9) is higher when using the blockchain by 8.73% (as expected due to the higher computational power required for blockchain operations), but we can't reject the null hypothesis in case of the memory (Figure 10), showing they might have similar memory usage patterns.

Table 3. CPU and memory usage for the SETS framework without blockchain (baseline).

Send Rate (m/s)	CPU (%)	Memory (GiB)
49.06	25.40	2058.31
76.13	26.36	1576.84
102.46	27.10	1608.57
120.96	28.04	1780.46

Table 4. CPU and memory usage for the SETS framework with blockchain.

Send Rate (m/s)	CPU (%)	Memory (GiB)
49.21	33.76	1696.54
72.69	35.08	1581.51
96.15	36.23	1484.62
120.04	36.78	1479.87

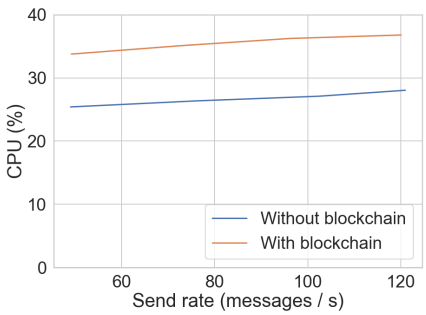


Figure 9. CPU usage comparison depending on the send rate.

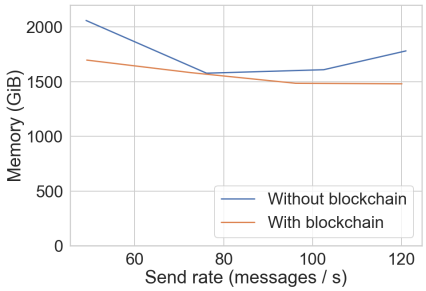


Figure 10. Memory comparison depending on the send rate.

6. Analysis

This section first analyses the SETS platform from a cyber resilience perspective, assessing how well the proposed design can mitigate potential threats. Next, it discusses how the SETS platform fulfils the evaluation criteria introduced in Section 2.2.2.

6.1. Potential Attacks

This section discusses how the developed CTI sharing platform eliminates the potential threats that any party might launch against it. The potential attacks are analyzed based on the model presented in [50].

1. **General public**: Anyone from an organization, state or hacker may play the role of an adversary in a general public category. A malicious actor is motivated by financial rewards, which it may gain directly or indirectly from mounting a certain attack, and would alter the message describing the respective incident with the hope that either a CTI consumer or the CTI sharing platform would receive the invalid information and would further act upon it. In both cases, the SETS framework and potential CTI consumers, before taking into use the threat information, need to examine the integrity of data using the hash persisted on the private blockchain by a CTI provider. Thereby, they can easily spot any changes in the received data and discard them. To modify the hash stored on the private blockchain in a way that would match the changes in the message, an adversary would need the same permission level as the CTI provider, which would be difficult considering the nature of the private blockchain. Furthermore, all transactions would be logged on the blockchain and later could be utilized for tracking down the blockchain breach.
   An attacker can also target the authenticity of data records in the SETS platform. To achieve this, the attacker needs to take over control of the backend plugin and subsequently modify the content of the records. The platform, however, would find out about the incident as soon as it retrieves the record in response to a consumer's query since it checks the integrity of data after each retrieval against the hash on the private blockchain and invalidates the data record in case any hash mismatch observed.
2. **Security expert**: A security expert may produce information about the launched attacks, attack patterns and attacker behaviours. On the other hand, the expert may also subscribe to threat information generated by other organizations. As stated in Section 4.2, only authorized experts can interact with the SETS platform. Additionally, an expert would need the appropriate permissions to be able to read or store hashes associated with CTI data on the private blockchain. Thereby, it is relatively challenging for an adversary to masquerade as a genuine expert and thus would be given the permissions for accessing the SETS platform and the private blockchain. Alternatively, compromising the expert's system would take place on two levels: stealing credentials for authorization on the SETS platform and acquiring cryptographic primitives for accessing the private blockchain, which would not be easy to achieve.
3. **Platform operator**: In this work, it is assumed that all decisions regarding membership and access rights would be made by a trustworthy third party designated as a platform operator who is responsible for governing the SETS platform. Additionally, the

operator is legitimized by a group of trusted third parties such as EU FI-ISAC [51]. Consequently, CTI providers and consumers could simply verify the authenticity of the platform operator from the consortium of trusted third parties before initiating any data exchange. Alternatively, an attacker trying to impersonate a genuine operator would need to compromise most parties in the consortium which would not be an easy task considering the stringent security measures placed in such organizations [8].

6.2. Qualitative Assessment of The Evaluation Criteria

This section addresses the second research question of this paper which was about the trade-offs between trust, immutability and privacy in the threat-sharing process.

6.2.1. Integrity

Data integrity is guaranteed by our addition of a blockchain component in the CTI sharing process. More specifically, as messages are published to the SETS platform and their hashes along with the messages' metadata are, at the same time, written to a private blockchain instance, to which only authorised parties have write access, any message received by a consumer can and should be verified before acting due to the gained information. This verification needs only read access to the private blockchain, therefore there is a light permission distribution, while the advantage is significant.

6.2.2. Performance

Our quantitative performance analysis reveals that the addition of a blockchain in the consumption process increases latency by an average of 3.05 s and reduces throughput by 8.87 messages per second. The performance is constant no matter the input send rate, at a message consumption success rate of over 95%, showing that the bottleneck is not due to the load. Combined with the fact that the throughput in the baseline implementation, without a blockchain, is capped at 10 messages per second, which is 5 to 13 times lower than the input send rate, we can be confident when saying that the baseline SETS framework (which is the official TAXII server with a minimally viable implementation) needs to be improved for a much higher event processing rate, before reassessing other features added to the system. The resource utilisation of the sharing framework increases when adding blockchain for integrity, but the increase is an acceptable margin. We conclude that our solution has a stable performance and success rate, while the major performance loss incurs in the baseline implementation of the TAXII server which has the potential to be improved.

6.2.3. Trust

Trust is the core of our solution—all relevant parties, from the **platform operator** to the security **experts** interested in data sharing have to pass through a robust identification process and only then they will receive the necessary permission level to both the private blockchain (not all parties need write access and, generally, no party should have the power to modify written data) and the sharing platform. These permissions are given to institutional actors through a real-life identification and authorisation process from inter-state actors and without them, no operations can be successfully executed. Moreso, trust is gained by cross-checking between the hashes written in the blockchain, the messages recorded by the sharing platform's server and the various logs managed by the platform operator and at the level of the blockchain transactions.

6.2.4. Privacy

Privacy is the main concern in data-sharing systems that are currently active, and our proposed system considers that from the **design stage**. The threat data which has the potential to reveal identifiable information about the threat or the threat's publisher is stored privately by the SETS platform using its backend plugins, where the general public cannot access it. This identifiable data can be later invalidated or completely removed after a relevancy period, as to respect the right to be forgotten. The only pieces of information stored

in an immutable way are hashes of the messages and of the messages' metadata (which include CTI object ids for identification purposes), which are information-compressing and first pre-image resistant, so it is especially difficult to reconstruct message information from the hash itself, making the identifiable data, for all intents and purposes, anonymous. Moreover, having the metadata part of the hashed message protects the SETS system from a collision attack, when a second message is found that ends up in the same hash (this kind of attack would need to be combined with overriding the server storage to control the metadata).

6.2.5. Automation

The addition of the publish-subscribe middleware checks the automation criteria: upon reception of a new event, the sharing platform server forwards the message to the specific queue designated for the relevant topic in pub-sub middleware (or multiple queues, if necessary), wherefrom all consumers which are subscribed to the topic are notified and get to consume the message independently. Consequently, the affected organizations would access threat information immediately after its discovery and without taking any manual action, thus meeting the target of automation. The message is published in its unverified state, which is why all parties must first validate any message themselves using the blockchain before taking further action using the gained information. To prevent data alterations at the server's storage level at the moment of following data queries, the integrity of the message would be automatically verified on both the server and inquirer to ensure validity. In case the server itself detects a mismatch, it will mark the data as altered and will reject the following queries for that specific data.

7. Discussion

The solution presented in this paper addresses the key challenges of threat sharing as follows: establishing trusted relationships between participants through robust authentication and authorization method, guaranteeing the reliability of data via integrity check, fostering interoperability via unified semantics and message formats, preserving data privacy by persisting data hashes on the blockchain and automating all processes pertain to data exchange. Thereby, this work does not suffer from the flaws such as lack of data privacy [32] and reliability [32] that state-of-art blockchain-based solutions did. Furthermore, the SETS platform is relatively straightforward to deploy because it has been fully Dockerized and can be simply run as several Docker services. As shown in Section 5.2, these services are resource-efficient, particularly in terms of CPU and memory.

Widespread deployment of SETS systems across national and international organizations would have twofold advantages: firstly, the time between breach and sharing information about the incident would be reduced substantially because security experts would leverage SETS to deliver threat information to the concerned parties promptly. As a consequence, this approach minimizes the destructive impacts of cyberattacks which would typically lead to affecting the next victims within a short period (e.g., an hour) [52]. Secondly, the authenticity of CTI in transition can be simply examined at any phase of the sharing process, thus the expert offering information could ensure that the intended data is securely delivered to the parties with similar interests. Consequently, organizations can mitigate complex cyberattacks and disastrous cascading effects using reliable data which was not plausible without the extensive SETS deployment.

In the developed solution, publishing threat information by an expert triggers the automated procedure wherein a message is relayed to SETS and its hash is written to the blockchain. Next, SETS distributes the data to the subscribed parties and, eventually, the integrity of the message is verified by both SETS and potential consumers. All the above transactions are carried out without any human intervention. Thereby, the solicited or unsolicited human errors would be eliminated, leading to a higher degree of reliability and security associated with threat sharing process.

Despite the benefits of implementing SETS with TAXII and blockchain, the prototype inherits their shortcomings as well [43,53,54]. As shown in experiment results (Section 5.2), the official TAXII framework is relatively slow in relaying incoming messages to potential consumers. These inefficiencies might be attributed to the Flask web framework, the key component of the TAXII prototype, which is not suitable for the production environment as a result of its inherited deficiencies namely lack of support for advanced message handling and limited security features and thus needs to be replaced with a more efficient and secure web server such as Gunicorn [55].

As shown in Section 5.2, adding blockchain support to the TAXII official implementation degrades the performance of the overall sharing process even more. This is the price participants would pay to exchange data securely while preserving data privacy. The increased latency is on a scale of a few seconds (average 3.02 s), which is reasonable given the fact experts typically need a few hours or even days to design and implement appropriate countermeasures.

Blockchain technology plays a key role in the SETS platform, thus the choice of blockchain would have a major impact on the system architecture. For instance, if the system favours more trust and security over privacy and performance it might opt for public blockchains such as Ethereum and Bitcoin but if its requisites are a fully closed execution environment, high performance and private transactions, it may go for private permissioned blockchains such as Hyperledger Fabric and Quorum. As this prototype utilizes the blockchain for hash storage aimed at integrity verification and allows only the authorized organizations to join the blockchain network for transaction executions, thus SETS includes the adapters for Hyperledger Fabric and Quorum. It is worth noting that, for evaluation purposes and without loss of generality, SETS has been configured with the Quorum network.

8. Conclusions

This paper presents a solution that turns the threat-sharing experience into a more effective and efficient one, compared to the existing methods. The design of the solution, along with the prototype implementation that combines the official TAXII framework and the private blockchain, is described in great detail. Additionally, the prototype is evaluated from different perspectives, namely performance and cyber resilience, validating the applicability and reliability of the solution, particularly for real-life use cases.

The state-of-art threat-sharing platforms that leverage blockchain technology either persist full data records in public blockchains and thus trade performance and data privacy for a higher degree of trust or store information in private blockchains, addressing performance issues associated with public blockchains but undermining the GDPR rule concerning the right to be forgotten. The solution presented in this paper addresses all the above problems by leveraging the database and the private permissioned blockchain as storage for data records and hashes, respectively. Therefore, the SETS platform improves the performance of the threat-sharing process through the execution of data manipulation operations such as creating, updating, and deleting within databases, and limiting interactions with the blockchain to writing and reading hash objects. In SETS, only the authorized parties can actively engage in exchanging data and further verify the integrity of the shared information using hash objects on the blockchain. According to the experimental outcomes, the average delay that organizations experience due to integrity checks supported by SETS is reasonably low, particularly considering the high costs of cyberattacks.

In future work, the sluggishness of SETS implementation that spawns from the official TAXII prototype can be enhanced by incorporating a more powerful web server. Additionally, the SETS framework similarly to several data marketplaces which were presented in [56] can develop incentive models to address asymmetric relationships between CTI providers and consumers.

Author Contributions: M.P. designed a blockchain-enabled Secure and Efficient Threat Sharing platform, developed the proof-of-concept and conducted the research; V.I. set up the experimental environment and carried out several test cases. He also evaluated the empirical results. Both authors contributed to the writing of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the EU H2020 PHOENIX project, Contract no. 832989.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available since the presented framework is part of the bigger platform which is partially EU-restricted.

Acknowledgments: I would like to express my great appreciation to Pasi Lassila for proofreading and his invaluable comments on an earlier version of the manuscript that greatly enhanced the manuscript.

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

References

1. DHS, US. Critical Infrastructure Sectors. 2019. Available online: https://www.cisa.gov/critical-infrastructure-sectors (accessed on 10 April 2022).
2. Digital Agenda for Europe, COM(2010)245 Final. 2010. Available online: https://www.eumonitor.eu/9353000/1/j9vvik7m1c3gyxp/vikqhod6cfud (accessed on 10 February 2022).
3. Onyeji, I.; Bazilian, M.; Bronk, C. Cyber security and critical energy infrastructure. *Electr. J.* **2014**, *27*, 52–60. [CrossRef]
4. Kokkonen, T.; Hautamäki, J.; Siltanen, J.; Hämäläinen, T. Model for sharing the information of cyber security situation awareness between organizations. In Proceedings of the 2016 23rd International Conference on Telecommunications (ICT), Thessaloniki, Greece, 16–18 May 2016; pp. 1–5.
5. Leszczyna, R.; Łosiński, M.; Małkowski, R. Security information sharing for the polish power system. In Proceedings of the 2015 Modern Electric Power Systems (MEPS), Wroclaw, Poland, 6–9 July 2015; pp. 1–6.
6. Johnson, C.; Badger, L.; Waltermire, D.; Snyder, J.; Skorupka, C. Guide to cyber threat information sharing. *NIST Spec. Publ.* **2016**, *800*, 150.
7. Martínez, M.M.; Marin-Tordera, E.; Masip-Bruin, X. Scalability analysis of a blockchain-based security strategy for complex IoT systems. In Proceedings of the 2021 IEEE 22nd International Conference on High Performance Switching and Routing (HPSR), Paris, France, 7–10 June 2021; pp. 1–6.
8. Allouche, Y.; Tapas, N.; Longo, F.; Shabtai, A.; Wolfsthal, Y. TRADE: TRusted Anonymous Data Exchange: Threat Sharing Using Blockchain Technology. *arXiv* **2021**, arXiv:2103.13158.
9. Pahlevan, M.; Voulkidis, A.; Velivassaki, T.H. Secure exchange of cyber threat intelligence using TAXII and distributed ledger technologies-application for electrical power and energy system. In Proceedings of the 16th International Conference on Availability, Reliability and Security, Vienna, Austria, 17–20 August 2021; pp. 1–8.
10. Tokarski, M. Protection of Individuals in the light of EU Regulation 2016/679 on the Protection of Natural Persons with Regard to the Processing of Personal Data and on the Free Movement of such Data. *Saf. Def.* **2020**, *6*, 63–74. [CrossRef]
11. Wang, X.; Zha, X.; Ni, W.; Liu, R.P.; Guo, Y.J.; Niu, X.; Zheng, K. Survey on blockchain for Internet of Things. *Comput. Commun.* **2019**, *136*, 10–29. [CrossRef]
12. Kuo, T.T.; Kim, H.E.; Ohno-Machado, L. Blockchain distributed ledger technologies for biomedical and health care applications. *J. Am. Med. Inform. Assoc.* **2017**, *24*, 1211–1220. [CrossRef]
13. Mendez Mena, D.; Yang, B. Decentralized Actionable Cyber Threat Intelligence for Networks and the Internet of Things. *IoT* **2021**, *2*, 1–16. [CrossRef]
14. Bissell, K.; Lasalle, R.M.; Dal Cin, P. The Cost of Cybercrime—Ninth Annual Cost of Cybercrime Study. Ponemon Institute and Accenture Security. 2019; Volume 50. Available online: https://www.accenture.com/_acnmedia/PDF-96/Accenture-2019-Cost-of-Cybercrime-Study-Final.pdf (accessed on 10 August 2022).
15. Luiijf, H.; Kernkamp, A. *Sharing Cyber Security Information: Good Practice Stemming from the Dutch Public-Private-Participation Approach*; TNO: The Hague, The Netherlands, 2015.
16. Brown, S.; Gommers, J.; Serrano, O. From cyber security information sharing to threat management. In Proceedings of the 2nd ACM Workshop on Information Sharing and Collaborative Security, Denver, CO, USA, 12 October 2015; pp. 43–49.
17. Dandurand, L.; Serrano, O.S. Towards improved cyber security information sharing. In Proceedings of the 2013 5th International Conference on Cyber Conflict (CYCON 2013), Tallinn, Estonia, 4–7 June 2013; pp. 1–16.
18. Haass, J.C.; Ahn, G.J.; Grimmelmann, F. ACTRA: A case study for threat information sharing. In Proceedings of the 2nd ACM Workshop on Information Sharing and Collaborative Security, Denver, CO, USA, 12 October 2015; pp. 23–26.
19. Skopik, F.; Settanni, G.; Fiedler, R. A problem shared is a problem halved: A survey on the dimensions of collective cyber defense through security information sharing. *Comput. Secur.* **2016**, *60*, 154–176. [CrossRef]
20. Jasper, S.E. US cyber threat intelligence sharing frameworks. *Int. J. Intell. Counterintell.* **2017**, *30*, 53–65. [CrossRef]

21. Tounsi, W.; Rais, H. A survey on technical threat intelligence in the age of sophisticated cyber attacks. *Comput. Secur.* **2018**, *72*, 212–233. [CrossRef]
22. Wagner, C.; Dulaunoy, A.; Wagener, G.; Iklody, A. Misp: The design and implementation of a collaborative threat intelligence sharing platform. In Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security, Vienna, Austria, 24 October 2016; pp. 49–56.
23. Grønberg, M. An Ontology for Cyber Threat Intelligence. Master's Thesis, University of Oslo, Oslo, Norway, 2019.
24. Mandiant. OpenIOC. 2010. Available online: http://www.openioc.org/ (accessed on 10 February 2022).
25. MITRE. Cyber Observable eXpression. 2011. Available online: https://cybox.mitre.org/about/ (accessed on 10 February 2022).
26. Barnum, S. Standardizing cyber threat intelligence information with the structured threat information expression (stix). *Mitre Corp.* **2012**, *11*, 1–22.
27. Connolly, J.; Davidson, M.; Schmidt, C. *The Trusted Automated Exchange of Indicator Information (Taxii)*; The MITRE Corporation: McLean, VA, USA, 2014; pp. 1–20.
28. Yli-Huumo, J.; Ko, D.; Choi, S.; Park, S.; Smolander, K. Where is current research on blockchain technology?—A systematic review. *PLoS ONE* **2016**, *11*, e0163477. [CrossRef] [PubMed]
29. Taylor, P.J.; Dargahi, T.; Dehghantanha, A.; Parizi, R.M.; Choo, K.K.R. A systematic literature review of blockchain cyber security. *Digit. Commun. Netw.* **2020**, *6*, 147–156. [CrossRef]
30. Homan, D.; Shiel, I.; Thorpe, C. A new network model for cyber threat intelligence sharing using blockchain technology. In Proceedings of the 2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Canary Islands, Spain, 24–26 June 2019; pp. 1–6.
31. Thummavet, P. Demystifying Hyperledger Fabric (1/3): Fabric Architecture. 2019. Available online: https://medium.com/coinmonks/demystifying-hyperledger-fabric-1-3-fabric-architecture-a2fdb587f6cb (accessed on 14 April 2022).
32. Purohit, S.; Calyam, P.; Wang, S.; Yempalla, R.; Varghese, J. DefenseChain: Consortium Blockchain for Cyber Threat Intelligence Sharing and Defense. In Proceedings of the 2020 2nd Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS), Paris, France, 28–30 September 2020; pp. 112–119.
33. Hajizadeh, M.; Afraz, N.; Ruffini, M.; Bauschert, T. Collaborative cyber attack defense in SDN networks using blockchain technology. In Proceedings of the 2020 6th IEEE Conference on Network Softwarization (NetSoft), Ghent, Belgium, 29 June–3 July 2020; pp. 487–492.
34. Kreutz, D.; Ramos, F.M.; Verissimo, P.E.; Rothenberg, C.E.; Azodolmolky, S.; Uhlig, S. Software-defined networking: A comprehensive survey. *Proc. IEEE* **2014**, *103*, 14–76. [CrossRef]
35. Magdziarczyk, M. Right to Be Forgotten in Light of Regulation (Eu) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the Protection of Natural Persons with Regard to the Processing of Personal Data and on the Free Movement of Such Data, and Repealing Directive 95/46/Ec. In Proceedings of the 6th International Multidisciplinary Scientific Conference on Social Sciences and Art SGEM 2019, Vienna, Austria, 24 August–2 September 2019; pp. 177–184.
36. Büber, E.; Şahingöz, Ö.K. Blockchain Based Information Sharing Mechanism for Cyber Threat Intelligence. *Balk. J. Electr. Comput. Eng.* **2020**, *8*, 242–253. [CrossRef]
37. Wu, Y.; Qiao, Y.; Ye, Y.; Lee, B. Towards improved trust in threat intelligence sharing using blockchain and trusted computing. In Proceedings of the 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS), Granada, Spain, 22–25 October 2019; pp. 474–481.
38. Cha, J.; Singh, S.K.; Pan, Y.; Park, J.H. Blockchain-Based Cyber Threat Intelligence System Architecture for Sustainable Computing. *Sustainability* **2020**, *12*, 6401. [CrossRef]
39. OASIS. STIX TM Version 2.0. Part 1: STIX Core Concepts. 2017. Available online: http://docs.oasis-open.org/cti/stix/v2.0/cs01/part1-stix-core/stix-v2.0-cs01-part1-stix-core.html (accessed on 15 March 2022).
40. Leach, P.; Mealling, M.; Salz, R. *A Universally Unique Identifier (uuid) urn Namespace*; Technical Report; The Internet Society: Reston, VA, USA, 2005.
41. OASIS. cti-taxii-server, 2017. Available online: https://github.com/oasis-open/cti-taxii-server (accessed on 10 February 2022).
42. OASIS. cti-taxii-client, 2017. Available online: https://github.com/oasis-open/cti-taxii-client (accessed on 10 February 2022).
43. Baliga, A.; Subhod, I.; Kamat, P.; Chatterjee, S. Performance evaluation of the quorum blockchain platform. *arXiv* **2018**, arXiv:1809.03421.
44. Project, P. Flask-Web Development, One Drop at a Time. 2010. Available online: https://flask.palletsprojects.com/en/1.1.x/ (accessed on 15 April 2022).
45. Synelixis. Open Source Identity and Access Management for Modern Applications and Services. 2021. Available online: https://www.keycloak.org/ (accessed on 16 March 2022).
46. Sollfrank, M.; Loch, F.; Denteneer, S.; Vogel-Heuser, B. Evaluating docker for lightweight virtualization of distributed and time-sensitive applications in industrial automation. *IEEE Trans. Ind. Inform.* **2020**, *17*, 3566–3576. [CrossRef]
47. Pure Python RabbitMQ/AMQP 0-9-1 Client Library. 2022. Available online: https://github.com/pika/pika (accessed on 17 April 2022).
48. Boskamp, E. 29 Worrisome Cybersecurity Statistics. 2022. Available online: https://www.zippia.com/advice/cybersecurity-statistics/ (accessed on 10 February 2022).

49. Mazzoni, M.; Corradi, A.; Di Nicola, V. Performance evaluation of permissioned blockchains for financial applications: The ConsenSys Quorum case study. *Blockchain Res. Appl.* **2022**, *3*, 100026. [CrossRef]
50. Satija, S.; Mehra, A.; Singanamalla, S.; Grover, K.; Sivathanu, M.; Chandran, N.; Gupta, D.; Lokam, S. Blockene: A high-throughput blockchain over mobile devices. In Proceedings of the 14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20), Virtual Event, 4–6 November 2020; pp. 567–582.
51. European Union Agency for Network and Information Security (ENISA). Information Sharing and Analysis Centres (ISACs) Cooperative Models. 2017. Available online: https://www.enisa.europa.eu/publications/information-sharing-and-analysis-center-isacs-cooperative-models (accessed on 10 February 2022).
52. Verizon RISK Team. 2015 Data Breach Investigations Report. 2015. Available online: https://old.iktissadevents.com/files/media/speeches/ACCF-2015-S4-lorenz-kuhlee.pdf (accessed on 10 February 2022).
53. Andola, N.; Gogoi, M.; Venkatesan, S.; Verma, S. Vulnerabilities on hyperledger fabric. *Pervasive Mob. Comput.* **2019**, *59*, 101050. [CrossRef]
54. Yamashita, K.; Nomura, Y.; Zhou, E.; Pi, B.; Jun, S. Potential risks of hyperledger fabric smart contracts. In Proceedings of the 2019 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE), Hangzhou, China, 24 February 2019; pp. 1–10.
55. Gunicorn. Green Unicorn. Available online: https://gunicorn.org/ (accessed on 10 February 2022).
56. Christidis, J.; Karkazis, P.A.; Papadopoulos, P.; Leligou, H.C.N. Decentralized Blockchain-Based IoT Data Marketplaces. *J. Sens. Actuator Netw.* **2022**, *11*, 39. [CrossRef]