DOI: 10.15439/2023F6880
ISSN 2300-5963 ACSIS, Vol. 35

Standards-based Cyber Threat Intelligence sharing using private Blockchains

Kimonas Provatas
National Technical University of Athens and IBM Hellas
NTUA Campus, Zografos 15780, Greece
Email: kimonaspro99@gmail.com

Ioannis Tzannetos
0009-0009-7505-965X
National Technical University of Athens, Software Engineering Lab, NTUA Campus, Zografos 15780, Greece
Email: itzannetos@mail.ntua.gr

Vassilios Vescoukis
0000-0002-5360-8349
National Technical University of Athens, Software Engineering Lab, NTUA Campus, Zografos 15780, Greece
Email: v.vescoukis@cs.ntua.gr

***Abstract*—As cyber-attacks become more and more sophisticated, sharing information that helps organizations design and implement efficient defense measures, is of critical importance. Such information can be shared using any service available, such as plain-old mailing lists, forums, etc. More mature systems use standards that facilitate the structural and semantic organization of information about cyber threats, which enables both automated processing and interpretation of such info, such as indexing, cross-referencing, updating, and more. However, even systems sharing cyber-attack info are themselves vulnerable, not only to typical and easily detectable attacks such as DoS, but also to content poisoning. Implementing such systems using decentralized architectures such as Blockchain, could overcome many deficits of centralized cyber-threat-info sharing systems. This paper presents the specification, design and implementation of such a decentralized system using two popular standards for cyber threat intelligence sharing, namely STIX for representing and TAXII for sharing such info using a REST API. The system, implemented on Hyperledger Fabric, faces the challenge of adhering to standards designed for a centralized world, and offering a transparent way for implementing all the backend, on a Blockchain.**

***Index Terms*— Blockchain, Cyber Threat Intelligence, Cyber defense, TAXII**

I. Introduction

IN THE field of cybersecurity, attackers and defenders are in a constant battle to outdo each other. Obtaining data about attackers' methods, tools, targeted vulnerabilities etc., support defenders in predicting attack targets and patterns, which is critical to proactively adjusting defenses, developing awareness and even preventing future attacks. Cybersecurity threat intelligence is the process of collecting appropriate cybersecurity data, evaluating it in the general context of its source and reliability, and analyzing it with methodical and structured techniques by specialized personnel, in the context of each organization. Collecting Cybersecurity Threat Intelligence (CTI) is a cyclical continuous process that employs several techniques, such as automation to extract only relevant information from data sources, human intervention by experts to understand and analyze information about threats and attack patterns, as well as integration with existing cybersecurity systems [1]. Considering that CTI is of great value, it is itself critical and must be trusted and dependable. To support this process and facilitate secure CTI exchange, two standards have been introduced: Structured Threat Information Expression (STIX) is a language and serialization format used to represent CTI data elements [2]; Trusted Automated Exchange of Intelligence Information (TAXII) is an application protocol for securely exchanging CTI over HTTPS. TAXII defines a RESTful API (a set of web services and message exchange services) and a set of requirements for TAXII Clients and Servers [3]. Even though it is expected CTI-sharing services to be offered over high-security infrastructures, it remains true that centralized implementations of such services, whose security is based on traditional centralized concepts, suffer themselves from vulnerabilities inherent to all centralized systems. Motivated by the challenge to further improve the security of CTI-sharing services, in this paper we investigate the benefits of providing CTI over a decentralized Blockchain infrastructure. We propose an architecture of a Threat Intelligence sharing service that implements the STIX/TAXII standards over a private permissioned Blockchain running on the Hyperledger Fabric network, instead of a centralized client-server model, to exploit the advantages of decentralized peer-to-peer trust models. Using a private Blockchain network such as Hyperledger Fabric can provide several advantages in a cybersecurity application as critical as sharing CTI. It can improve confidentiality by ensuring that only parties authorized by their trusted peers have access to the data; it also improves integrity by providing a tamper-proof record of all CTI producing and consuming transactions, availability by ensuring that time-critical access to CTI data does not depend on the availability of a central by-definition-trusted service, as well as non-repudiation by eliminating the possibility of denial of executed actions finally, it enhances auditability by providing a complete and transparent record of all in- and outbound CTI exchange transactions. We also present an implementation of the STIX/TAXII on Hyperledger Fabric and discuss observed advantages, issues and assumptions.

II. RELATED WORK

Aiming to support collaboration against threats, and in-line with EU legislation on information security, researchers have created a threat sharing system [12] using Hyperledger Fabric; the primary focus was towards addressing authorization concerns related to threat information. Authorization is accomplished using the native STIX traffic light protocol [13]. In other works [14], a threat sharing application was developed, motivated by the security properties offered by private blockchain and Hyperledger Fabric. The application was integrated with an SDN (Software-Defined Networking) Controller to exploit the synergy of threat intelligence and automation. Its primary objective was to enable seamless collaboration among organizations during distributed denial of service attacks and blacklist potentially malicious IP addresses during the flood, based on collective threat intelligence. In their study [15], the authors developed a CTI sharing platform tailored to the requirements of real-time threat intelligence in electrical power and energy systems. The platform comprised a generalized publish-subscribe middleware, which communicated with a Hyperledger Fabric network. Subsequently, the research was expanded [16] to tackle privacy concerns stipulated by GDPR (General Data Protection Regulation) and the performance overhead of storing large volumes of data on-chain. To accommodate this known issue in all Blockchains, they only stored the hash values of STIX objects on the Fabric Network, while storing the actual data on a separate database. Furthermore, the authors conducted both quantitative and qualitative analysis of the network's performance concerning various types of attacks. This work, although focused on threats in Energy systems, which is undoubtedly a critical domain, highlights the significance of strengthening cybersecurity and the growing interest in the development of advanced threat info sharing systems, leveraging technologies like Hyperledger Fabric and private permissioned Blockchains.

III. CTI Sharing on Blockchains: Requirements, Challenges and Advantages

Cyber Threat Intelligence is a challenging field, particularly in the context of multi-party collaboration, which clearly makes a lot of sense for both corporate and public sector cyber defense. Considering that CTI itself needs to be trusted and protected from malicious infections and alterations, the sharing of CTI among multiple organizations requires overcoming several challenges; the heterogeneity of data sources, the trustworthiness of data, the timely delivery of information, the need for privacy and confidentiality, as well as the availability of data even without network connections, are some of these challenges. Moreover, the accuracy and relevance of CTI are crucial for proactive defense against cyber threats, and is also very critical to be left upon centralized services, vulnerable or even malicious themselves. It is not uncommon CTI to shared within networks that, even if they are private, they still engage a centralized trust model. Therefore, establishing a zero-trust framework for collecting, analyzing, and sharing CTI among multiple parties is worth investigating. This framework should address both the technical and operational challenges of CTI sharing, while ensuring the protection of sensitive information and privacy, even from entities which are normally taken for trusted. Although standardization itself is a significant aspect of designing and developing information systems, this paper does not discuss the advantages of standardizing CTI sharing using STIX and TAXII. The focus is on the investigation of the benefits of utilizing a blockchain system to improve the security of organizations that are willing or are already a part of a Cyber Threat Intelligence (CTI) sharing network.

As several technology options for satisfying the above requirements may exist, in the sequel we will discuss the security properties that acted as selection criteria for a blockchain platform and how they are implemented using the Hyperledger Fabric mechanism.

- Organization level privacy: The TAXII standard requires confidentiality in STIX object collections, ensuring that only authorized organizations have read access to them. We address this requirement by utilizing private data collections on Hyperledger Fabric where actual transaction data is stored only in the nodes of organizations that have the required access, while others only receive metadata and hashes for the transaction [4].
- Organization level access control: The TAXII standard restricts the ability to write data to authorized organizations only, which is also satisfied by the Private Data Collections mechanism of Hyperledger Fabric [4]. However, within a conventional centralized client-server implementation of a TAXII Server, one single hosting organization has complete write authorization on all data stored in the database; this alone can be a deal-breaker for the participation of critical-mission strategic organizations (e.g. defense and civil protection bodies) in CTI sharing networks.
- Data Integrity: The TAXII standard does not impose a strong requirement or mechanism for verifying the integrity of the data, as this is out-of-scope of the standard. Nevertheless, it is considered necessary for any application in the field of cyber se-

curity to have a mechanism for data integrity verification. One can argue that organizations participating in a CTI sharing network generally rely on the information provided by other peers of the network, as they share a common goal; still, data integrity checks must be performed, since data tampering on the TAXII Server by malicious actors may lead to infected security information reaching all CTI consumers; thus, the integrity of shared CTI data constitutes a central point of failure. Being a central element of the nature of the Blockchain philosophy, this requirement is satisfied by all blockchain ecosystems, as data integrity needs to be verified and signed by all peers holding the information. [5]

- High Availability: For a CTI sharing application, it is crucial to ensure high availability. However, managing the TAXII Server, even within a single corporate network protected by firewalls, employing replicas etc, the single logical TAXII server is also a single point of failure. Malicious actors can launch various types of denial-of-service attacks that could render the server non-functional during critical times, such as attack campaign timeframes. Hyperledger Fabric offers a solution to this challenge by enabling organizations to manage multiple peers that provide redundancy of data and services at the organizational level, while still satisfying the data control and integrity requirements. The nodes of all organizations that participate in the network maintain a copy of the distributed ledger at all times [5]. This ensures that access of other peers to CTI can be provided, even if all nodes of one organization become for any reason unavailable.
- Non Repudiation and Auditability: Injection of incorrect or even malicious cybersecurity information from one member to the CTI sharing network, can have catastrophic implications for other members. In such a case, in centralized systems governed by one single entity, it cannot be guaranteed that the information producer will be charged for its erroneous or malicious activities so as to be rendered responsible. However, the TAXII Server administrator(s) should not be able to take any action that protect any peer from taking the responsibility of its mistakes or malicious activities. The inherent feature of decentralized transaction write-only transaction ledgers been maintained by all Blockchain nodes of all organizations, typically satisfies the non-repudiation and auditability requirement.

Beyond the security properties achieved by migrating from a client server model to a blockchain platform there are significant platform specific advantages in Hyperledger Fabric compared to other blockchain networks.

- Hyperledger Fabric is a permissioned blockchain network. Compared to public blockchain networks such as Ethereum access of new members is strictly controlled and must be first approved by other participants [6]. We believe that this model serves better the purpose of threat intelligence sharing.
- Hyperledger Fabric offers feature-rich ways of interacting with the network using the Fabric SDK using general purpose programming languages and is designed with organization level decentralization in mind while public blockchains are designed for censorship resistance first. While these concepts mays seem similar at first, they are different.

As a conclusion, it is apparent that the development of a multiparty CTI sharing application on Hyperledger Fabric can potentially resolve several issues inherent in the traditional client-server model.

The primary challenge encountered by this paper is the commitment to adhere strictly to a STIX/TAXII standards implementation on a blockchain ecosystem. As these standards were originally designed to function on a RESTful API, the challenge arises from the need to translate them into an equivalent decentralized version. This means that to comply with the standard, HTTP requests must be used, and a REST API server must be integrated into the blockchain network. As mentioned above, all collaborating organizations will participate in the Blockchain network, each with at least one node; clients coming from each organization will be connected to their organization's node(s), as shown in Figure 1 below.

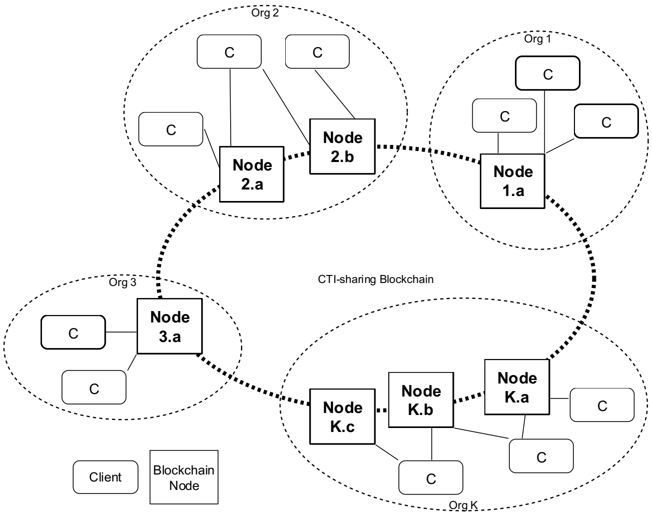


Figure 1. A Blockchain architecture for CTI sharing.

To address this, we need to enhance Hyperledger Fabric with a REST API server that acts as an intermediary to forward requests from the organization's clients to the blockchain network. This is not a bypass to the decentralized nature of the network, considering that it only acts as a relay or secondary client to the blockchain network; furthermore, in this architecture Hyperledger Fabric is decentralized only in the organization level, in contrast with Ethereum or other public blockchains that provide user level decentralization. The nature of this application allows organization-level decentralization, although user-level decentralization can still be possible in specific environments and applications.

Regarding the adherence to the STIX standard, a validator has been developed to verify STIX compliance; implemented as a proof of concept, it currently operates on the API server and is intended to be deployed as Fabric chaincode. This task was comparatively less challenging than integrating the TAXII server into the network, due to two reasons: Firstly, Fabric's data layer is based on a key-value store [7] (either LevelDB or CouchDB) that is inherently similar to JSON objects. Secondly, the validator can be tested off-chain and then effortlessly moved on-chain as it requires minimal or no additional blockchain-metadata to function properly. The work described in this paper encountered a final difficulty in selecting a scope for the API specification to design, implement, and document [8]. The TAXII standard is highly extensible [8], which made it challenging to identify the essential features that offer the basic functionality of threat intelligence sharing and facilitate system management, especially in scenarios involving multiple parties. The process of designing and implementing REST API endpoints that comply with the TAXII standard has been fully documented at various levels, using UML diagrams.

IV. System Design

As we mentioned above, the basic TAXII specification consists of a series of REST API endpoints [8]. The first step in order to design the system is to select a subset use cases to implement at the REST API level. These endpoints are divided into two categories, those that facilitate interaction with the blockchain that are completely custom, and those that provide the functionality required by the TAXII server standard; the API consumers should observe behavior identical to their client/server equivalent. This is shown as a use-case diagram in Figure 2.

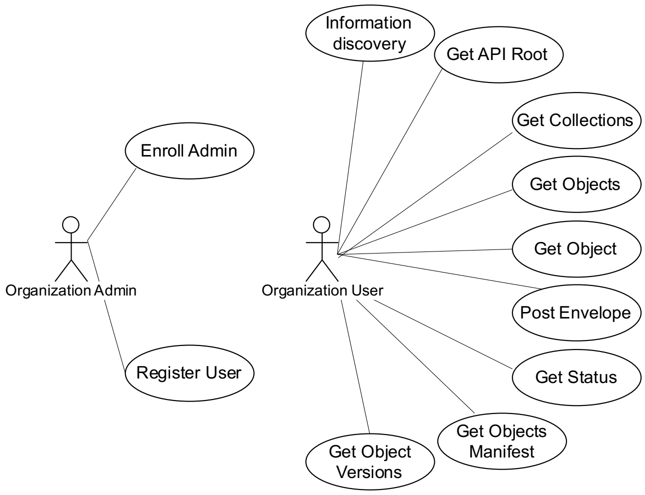


Figure 2. Roles and access to services offered.

In summary, based on the use case diagram, organization administrators are responsible for enrolling and registering users on the fabric-network [9]. Organization users can interact with every endpoint specified by the TAXII standard, with one critical exception: the ability to delete a STIX object from a collection. This is because blockchain transactions result in an immutable record of assets being transferred from nodes - CTI producers to other nodes. Thus, the best approach is to restrict access to assets beforehand using private data collections and access control. Once an asset reaches an organization's blockchain, it cannot be removed; this is a required behavior that allows the identification of the origin of false CTI coming from compromised or malevolent organizations.

The next step is to create an architecture containing an overview of the software components needed to implement the required functionalities. The UML component diagram below represents these components as containers or processes that live inside an organization's node; further details about their physical deployment will be discussed later. It is important to emphasize that this architecture needs to be implemented inside every single organization participating in the CTI-sharing Blockchain. Therefore, organizations may implement customized versions of this architecture in their production systems, such as fewer or more peer nodes, as discussed earlier; another parameter is whether one organization will be participating in the ordering service or not. Without compromising the principles of the proposed decentralized approach, we assume that every organization implements an identical infrastructure to standardize and simplify aspects of the blockchain layer that will be presented. Such an architecture is shown in Figure 3 below.

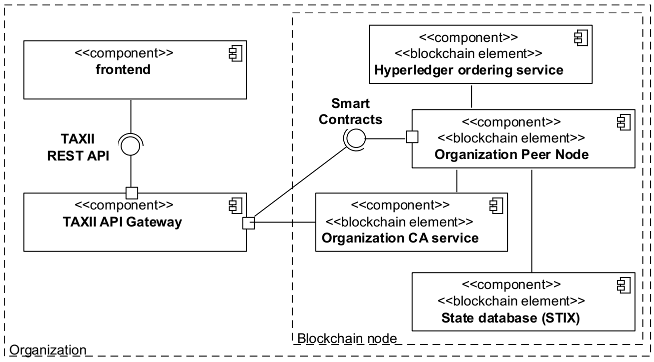


Figure 3. Component-level architecture of Blockchain nodes in organizations.

In the sequel we briefly discuss the system architecture shown in the above UML component diagram. The frontend component can be any implementation of a UI using typical web technologies (Apache, js), or a custom smart client that provides services such as event triggering upon incoming CTI etc.. In both cases, the frontend consumes the TAXII Rest API. The frontend needs not to be aware that the TAXII API offering is really based on a decentralized application. This allows even third-party TAXII clients to be used.

The Organization API Gateway is responsible for three main tasks: First, it validates the compliance of incoming requests with the TAXII standard, by checking HTTP headers, methods, and body parameters. It rejects any input that does not conform with the TAXII standard, without any further interaction with the blockchain backend. Second, it provides user authentication and authorization, by utilizing the organization's certification service(s). The API Gateway receives a token from the client and checks the organization's database for a corresponding certificate stored on behalf of this client. If the certificate exists, it is retrieved, and the request is relayed to the network with that certificate. If an authentication error occurs, the user is informed accordingly.

The third and most important function of the API Gateway is to submit to the blockchain-based backend, transactions on behalf of the organization's clients. These are the core of the TAXII server as they implement the main application business logic and ensure threat intelligence sharing functionality that benefits from the immutability of the blockchain. It is important to note that the transactions are not being run on the API Gateway itself: instead, they are submitted to the Hyperledger Fabric network to be executed on the Blockchain. The results are then received and forwarded to the actual clients. Further discussion on the smarts contracts will follow in the next section.

Consistent to the diagram of the software components, we present a UML deployment diagram to show the assignment of components to execution nodes. This deployment, shown in Figure 4, refers to a specific organization, and seems reasonable enough for production, as it physically isolates the REST API, the Fabric Infrastructure and the ordering service which are in fact very different in business functions.

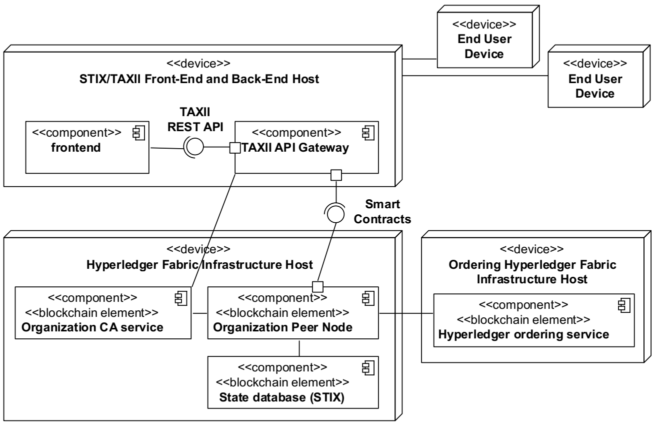


Figure 4. Deployment of the CTI Blockchain architecture.

Notably, a possible option for deploying the system is to combine the REST API and the fabric organization infrastructure into a single physical host. This approach may be beneficial for smaller organizations that want to conserve resources. However, it is advised to deploy the ordering nodes in separate physical hosts to improve security and performance. Furthermore, as mentioned earlier, it is possible that each organization maintains more than one blockchain node and ordering service, as shown in Figure 5 below.

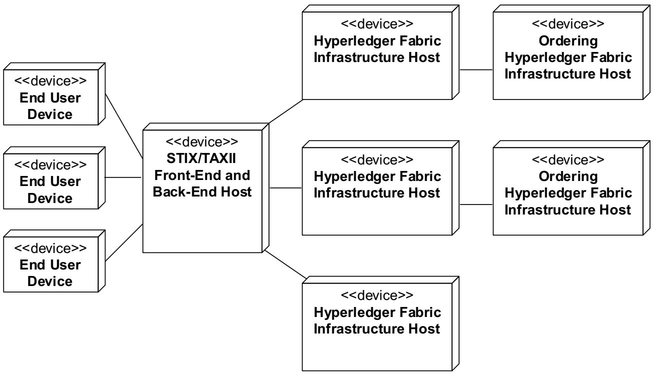


Figure 5. Alternative CTI Blockchain deployment for smaller organizations.

V. TECHNICAL IMPLEMENTATION

The main components that implement the blockchain are the peer nodes, which are the fundamental building blocks of the Hyperledger Fabric. These nodes serve two essential functions that comprise the blockchain network: ledger and transaction management. In the context of this paper, it is important to note that the peers serve as the hosting component for chaincode, which is a collection of smart contracts that are the main mechanism for interacting with the network [10]. Each peer node implements a number of smart contracts as shown in the UML class diagram in Figure 6 below.

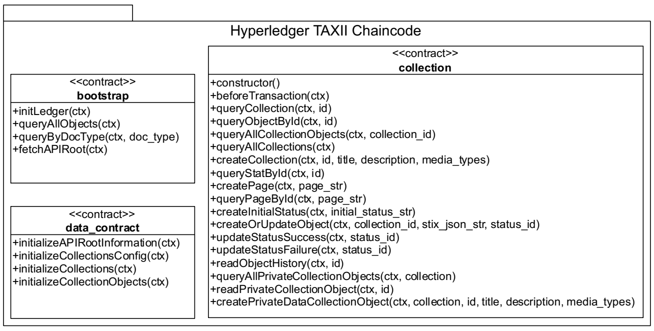
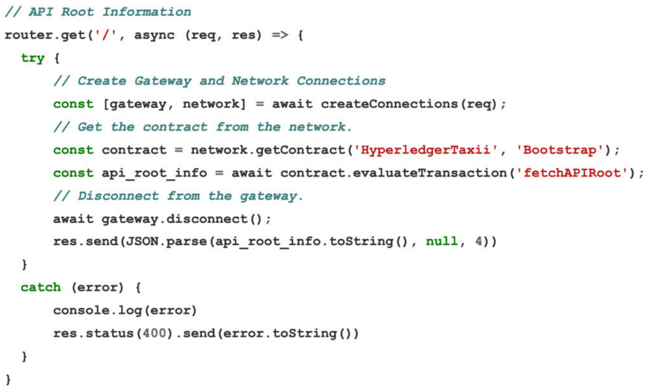

Hyperledger TAXII Chaincode
<<contract>>
bootstrap
+initLedger(ctx)
+queryAllObjects(ctx)
+queryByDocType(ctx, doc_type)
+fetchAPIRoot(ctx)
<<contract>>
data_contract
+initializeAPIRootInformation(ctx)
+initializeCollectionsConfig(ctx)
+initializeCollections(ctx)
+initializeCollectionObjects(ctx)
<<contract>>
collection
+constructor()
+beforeTransaction(ctx)
+queryCollection(ctx, id)
+queryObjectById(ctx, id)
+queryAllCollectionObjects(ctx, collection_id)
+queryAllCollections(ctx)
+createCollection(ctx, id, title, description, media_types)
+queryStatById(ctx, id)
+createPage(ctx, page_str)
+queryPageById(ctx, page_str)
+createInitialStatus(ctx, initial_status_str)
+createOrUpdateObject(ctx, collection_id, stix_json_str, status_id)
+updateStatusSuccess(ctx, status_id)
+updateStatusFailure(ctx, status_id)
+readObjectHistory(ctx, id)
+queryAllPrivateCollectionObjects(ctx, collection)
+readPrivateCollectionObject(ctx, id)
+createPrivateDataCollectionObject(ctx, collection, id, title, description, media_types)

Figure 6. Smartc contracts implemented in peer nodes.

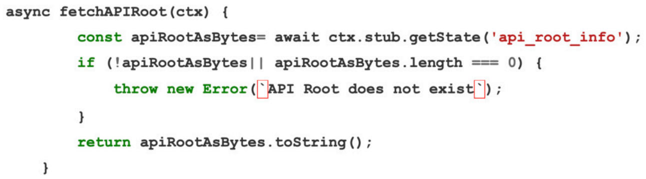
The diagram depicts a chaincode called Hyperledger TAXII Chaincode, which is installed on the file system of the organization's peer node. This chaincode implements three smart contracts: bootstrap, data_contract, and collection. The bootstrap and data_contract contracts are standard patterns used to initialize the ledger and perform some administrative tasks that are later removed in production. It is the collection contract that handles the core application logic of the TAXII server and is responsible for every task related to TAXII terminology, from API Root information to STIX object reads and writes. In Figure 7 below we list an example of the most basic REST API call in javascript code at HTTP server detail:



```
// API Root Information
router.get('/', async (req, res) => {
  try {
      // Create Gateway and Network Connections
      const [gateway, network] = await createConnections(req);
      // Get the contract from the network.
      const contract = network.getContract('HyperledgerTaxii', 'Bootstrap');
      const api_root_info = await contract.evaluateTransaction('fetchAPIRoot');
      // Disconnect from the gateway.
      await gateway.disconnect();
      res.send(JSON.parse(api_root_info.toString(), null, 4))
  }
  catch (error) {
      console.log(error)
      res.status(400).send(error.toString())
  }
}
```

Figure 7. JS code to implement a typical API call.

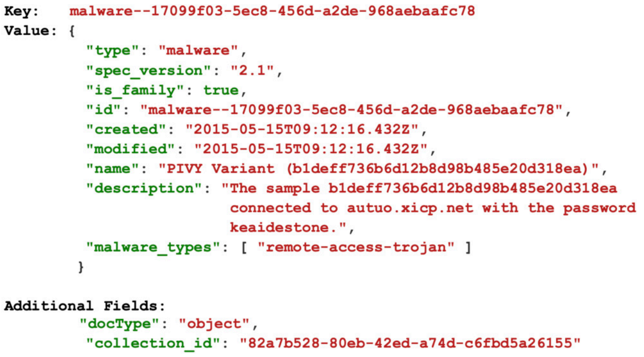
And at smart contract transaction level:



```
async fetchAPIRoot(ctx) {
        const apiRootAsBytes= await ctx.stub.getState('api_root_info');
        if (!apiRootAsBytes|| apiRootAsBytes.length === 0) {
            throw new Error(`API Root does not exist`);
        }
        return apiRootAsBytes.toString();
    }
```

Figure 8. JS code to implement a transaction.

The remaining three components to implement an instance of the system are the state database, certificate authority node, and ordering service. For the purposes of this work, the certificate authority node and ordering service remain unaltered from their default roles in Fabric. The state database serves as a cache for the current key-value pairs, known as the world state, in Hyperledger Fabric [7]. Past values are stored in the ledger. The state database was changed from Hyperledger's default levelDB, to couchDB for rich querying support and improved performance and management. It also stores certificates for organization users and CTI compliant data in STIX language [11] where a key of **<ObjectType>-<ID>** or **<ID>** format is used to quickly retrieve the STIX payload along with a custom field **<docType>**. In some cases a **<collection_id>** field is added and then stripped away before displaying to the user to identify the collection an object belongs to. These are standard best practices provided by fabric code examples. An object of type malware is stored inside the world state database in the format shown in Figure 9



```
Key:   malware--17099f03-5ec8-456d-a2de-968aebaafc78
Value: {
          "type": "malware",
          "spec_version": "2.1",
          "is_family": true,
          "id": "malware--17099f03-5ec8-456d-a2de-968aebaafc78",
          "created": "2015-05-15T09:12:16.432Z",
          "modified": "2015-05-15T09:12:16.432Z",
          "name": "PIVY Variant (b1deff736b6d12b8d98b485e20d318ea)",
          "description": "The sample b1deff736b6d12b8d98b485e20d318ea
                          connected to autuo.xicp.net with the password
                          keaidestone.",
          "malware_types": [ "remote-access-trojan" ]
       }

Additional Fields:
         "docType": "object",
          "collection_id": "82a7b528-80eb-42ed-a74d-c6fbd5a26155"
```

Figure 9. JSON representation for a STIX "malware" object.

The data flow within the system initiates with the API consumers who are permitted to access the system either via a web browser or by directly making requests to the REST API. The REST API server carries out authentication checks to verify the user's authorization to participate in the blockchain network. After the authentication is confirmed, the API server retrieves the user certificate and functions as a client of the blockchain network, submitting transactions on behalf of the user. In Figure 10 we demonstrate the UML sequence diagram that corresponds to GET API Root endpoint code above.

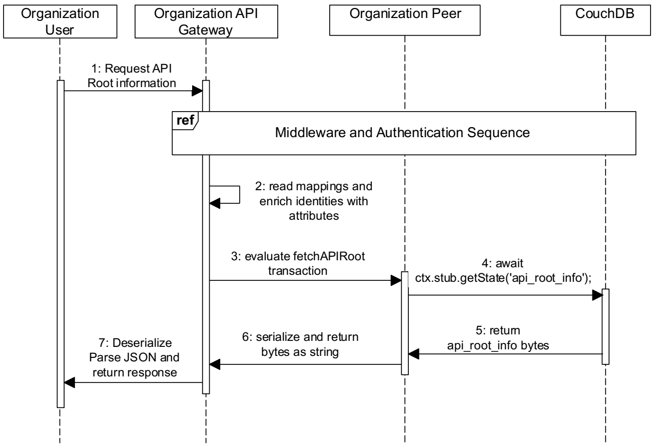

Organization User
Organization API Gateway
Organization Peer
CouchDB
1: Request API Root information
ref
Middleware and Authentication Sequence
2: read mappings and enrich identities with attributes
3: evaluate fetchAPIRoot transaction
4: await ctx.stub.getState('api_root_info');
5: return api_root_info bytes
6: serialize and return bytes as string
7: Deserialize Parse JSON and return response

Figure 10. A sequence diagram for the "GET API Root" endpoint.

In certain cases, an endpoint might require the submission of multiple transactions. However, submitting multiple transactions is not a recommended practice due to the potential inconsistencies and interruptions that may occur during the code execution. A more effective approach is to map each

REST API endpoint directly to a distinct Hyperledger Fabric transaction. This ensures that the transaction is not only clearly defined but can also be further expanded into sub-transactions as necessary. It is important to note, however, that the implementation of this principle is out of scope of this work, as it serves as a proof of concept.

VI. CONCLUSION

We introduce an architecture for implementing a decentralized Threat Intelligence sharing system utilizing the STIX/TAXII standard, over a Hyperledger Fabric network. The proposed system leverages blockchain technology to enhance security in existing threat sharing systems and standards. By implementing a private permissioned blockchain network like Hyperledger Fabric, the proposed system provides a range of advantages such as improved confidentiality, integrity, availability, non-repudiation, and auditability. The blockchain network ensures that only authorized parties have access to data, maintains a tamper-proof record of all transactions, guarantees that data is always accessible, prevents the denial of service, and provides a transparent and complete immutable record of all transactions. The proposed system uses a decentralized architecture, in which several components work together to provide seamless threat intelligence sharing functionality. These components include the front-end, organization API gateway, peer nodes, state database, certificate authority node, and ordering service. The decentralized architecture is designed to ensure that decentralization does not hinder the system's overall functionality. The proposed system employs REST APIs to facilitate end-users', including third-party TAXII clients interaction with the system, and smart contracts to execute transactions on the blockchain network. The results of this research demonstrate the feasibility and effectiveness of the proposed architecture for threat intelligence sharing systems.

Some future improvements to the functionality, security and performance of the overall system are as follows:

- More robust authentication scheme: In the context of security applications, the basic TAXII standard authentication method that employs the Authentication Basic scheme poses potential risks due to its simplicity. To mitigate these risks, we suggest as a future improvement using a more secure authentication scheme that combines Basic Authentication with Multi Factor Authentication to generate a new certificate. This approach significantly reduces the risk of unauthorized access, especially in cases where an attacker has the enrollmentID and enrollmentSecret elements. Using certificates instead of enrollmentID/enrollmentSecret pairs offers two significant advantages: it minimizes user credential exposure over the channel, even if encrypted with TLS, and the credentials are short-lived, which means they can be easily revoked through the Certificate Authority (CA) authority node if exposed to a malicious actor.
- Writing the STIX validator on chain: One way to ensure a common data format is to validate the STIX data using the validator before storing it in the TAXII Server. Currently, this validation is performed at the API Gateway level and at a basic level. However, this approach may allow organizations to deviate from the STIX standard after processing the Gateway code, which is off-chain. To address this, data validation can be performed at the peer nodes of the Hyperledger Fabric through Chaincode transactions, for more multi-party trust. This would solve the problem of organizations agreeing on a common data validation logic as the code would be common and visible to all. Additionally, the Fabric Chaincode Lifecycle process can change this code at any time in a manner that is agreed upon and approved by all organizations.
- Implementing voting functionality: In CTI systems, the anonymous exchange of threat intelligence often includes a reputation mechanism to incentivize sharing sensitive data with other analysts. However, in our proposed system, which is designed for smaller to medium-sized consortiums, such a mechanism is not necessary since it is not open to the public. However, a voting and/or ban mechanism implemented on the Blockchain may still be useful if the network grows beyond a certain size, to further prevent malicious activities such as data poisoning.

REFERENCES

[1] Cobb, M. and Wigmore, I. (2021) *What is threat intelligence (cyber threat intelligence)? – definition from whatis.com*, *WhatIs.com*. Available at: https://www.techtarget.com/whatis/definition/threat-intelligence-cyber-threat-intelligence

[2] *What is STIX?* (2020) *Introduction to stix*. Available at: https://oasis-open.github.io/cti-documentation/stix/intro.

[3] (2020) *Introduction to taxii*. Available at: https://oasis-open.github.io/cti-documentation/taxii/intro.html.

[4] *Private data* (2017) *hyperledger*. Available at: https://hyperledger-fabric.readthedocs.io/en/release-2.2/private-data/private-data.html.

[5] (2017) *Ledger*. Available at: https://hyperledger-fabric.readthedocs.io/en/release-2.2/ledger.html.

[6] *Hyperledger fabric network* (2017) *hyperledger*. Available at: https://hyperledger-fabric.readthedocs.io/en/release-1.2/network/network.html.

[7] *Hyperledger Fabric model* (2017) *hyperledger*. Available at: https://hyperledger-fabric.readthedocs.io/en/latest/fabric_model.html.

[8] *TAXII specification* (2020) *TAXII Version 2.1*. Available at: https://docs.oasis-open.org/cti/taxii/v2.1/os/taxii-v2.1-os.html.

[9] *Registering and enrolling identities with a CA* (2017) *hyperledger*. Available at: https://hyperledger-fabric-ca.readthedocs.io/en/latest/deployguide/use_CA.html.

[10] *Smart contracts and chaincode* (2017) *hyperledger*. Available at: https://hyperledger-fabric.readthedocs.io/en/latest/smartcontract/smartcontract.html.

[11] *STIX specification* (2020) *STIXTM Version 2.1*. Available at: https://docs.oasis-open.org/cti/stix/v2.1/csprd01/stix-v2.1-csprd01.html.

[12] A new network model for cyber threat intelligence sharing using

blockchain (2019). Available at: https://arrow.tudublin.ie/cgi/ viewcontent.cgi?article=1003&context=nsdcon

[13] Traffic Light Protocol (TLP) Definitions and Usage (2022). Available at https://www.cisa.gov/news-events/news/traffic-light-protocol-tlp-definitions-and-usage

[14] Collaborative Cyber Attack Defense in SDN Networks using Blockchain Technology (2020). Available at: https://www.researchgate.net/publication/343616521_Collaborative_Cyber_Attack_Defense_in_SDN_Networks_using_Blockchain_Technology.

[15] Secure exchange of cyber threat intelligence using TAXII and distributed ledger technologies - application for electrical power and energy system (2021). Available at: https://dl.acm.org/doi/10.1145/3465481.3470476

[16] Secure and Efficient Exchange of Threat Information Using Blockchain Technology (2022). Available at: https://www.mdpi.com/2078-2489/13/10/463