# CYBER THREAT INTELLIGENCE SHARING TECHNOLOGIES AND THREAT SHARING MODEL USING BLOCKCHAIN

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

AHMET ÖZDEMİR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
CYBER SECURITY

MAY 2021

Approval of the thesis:

**CYBER THREAT INTELLIGENCE SHARING TECHNOLOGIES
AND
THREAT SHARING MODEL USING BLOCKCHAIN**

submitted by **AHMET ÖZDEMİR** in partial fulfillment of the requirements for the degree of **Master of Science in Cyber Security Department, Middle East Technical University** by,

| | |
|---|---|
| Prof. Dr. Deniz Zeyrek Bozşahin<br>Dean, **Graduate School of Informatics** | ______________ |
| Assist. Prof. Dr. Cihangir Tezcan<br>Head of Department, **Cyber Security** | ______________ |
| Assist. Prof. Dr. Aybar Can Acar<br>Supervisor, **Medical Informatics, METU** | ______________ |
| Dr. Attila Özgit<br>Co-supervisor, **Cyber Security, METU** | ______________ |

**Examining Committee Members:**

| | |
|---|---|
| Prof. Dr. Ali Aydın Selçuk<br>Computer Engineering, TOBB ETU | ______________ |
| Assist. Prof. Dr. Aybar Can Acar<br>Medical Informatics, METU | ______________ |
| Assist. Prof. Dr. Cihangir Tezcan<br>Cyber Security, METU | ______________ |

**Date:** **7 May 2021**

## ABSTRACT

## CYBER THREAT INTELLIGENCE SHARING TECHNOLOGIES AND THREAT SHARING MODEL USING BLOCKCHAIN

Özdemir, Ahmet
M.Sc., Department of Cyber Security
Supervisor : Assist. Prof. Dr. Aybar Can Acar
Co-Supervisor : Dr. Attila Özgit

June 2021, 104 pages

Against the measures taken, the nature of the threats in the cyber environment is evolving day by day. While script kiddie made amateur cyber attacks were usually experienced beforehand, more sophisticated and targeted attacks are frequently encountered nowadays. Besides that, commonly used signature based techniques for attack detection and threat information staying within organization is insufficient for dynamically changing, organized and targeted threats. Furthermore, with the advance of new technology, computer networks are growing, the number and variety of interconnected devices are increasing and as a consequence attack surface is expanding. As a result, it does not seem possible to reduce all of the vulnerabilities that we encounter. From now on, cyber attack is not a matter of 'if', but it is a matter of 'when'. In order to detect complex attacks one of the newly developed approaches is cyber threat intelligence sharing. Threat intelligence is evidence-based knowledge about threat and assists to decide. It has no value if it is not disseminated though. Organizations can increase situational awareness about targeted cyber threats by sharing internal cyber threat information with trusted partners and integrating external cyber threat information with their security systems in real-time basis. However, common language that allows automation at identification and sharing of threat information is crucial for timely intervention. To that end, various standards are being developed by many organizations and companies. In this study, standards and tools developed for the representation and sharing of threat information are compared, and new threat sharing model is developed using a permissioned blockchain.

Keywords: cyber threat intelligence, information sharing, indicators of compromise (IOCs), blockchain

# ÖZ

## SİBER TEHDİT İSTİHBARAT PAYLAŞIM TEKNOLOJİLERİ VE BLOKZİNCİR KULLANILARAK OLUŞTURULAN TEHDİT PAYLAŞIM MODELİ

Özdemir, Ahmet
Yüksek Lisans, Siber Güvenlik Bölümü
Tez Yöneticisi : Dr. Öğr. Üyesi Aybar Can Acar
Ortak Tez Yöneticisi : Dr. Attila Özgit

Haziran 2021, 104 sayfa

Siber ortamdaki tehditlerin doğası alınan önlemler karşısında her geçen gün gelişmektedir. Önceleri 'script kiddie' tabir edilen siber saldırganlar tarafından gerçekleştirilen amatör saldırılar çoğunlukla yaşanırken, günümüzde daha karmaşık ve hedefli saldırılar sıklıkla karşımıza çıkmaktadır. Bunun yanında, saldırı tespitinde yaygın olarak kullanılan imza tabanlı yöntemler ve kurum içinde kalan tehdit bilgisi dinamik olarak değişen, organize olmuş ve hedefli tehditler karşısında yetersiz kalmaktadır. Ayrıca gelişen teknoloji ile birlikte bilgisayar ağları büyümekte, birbirine bağlı cihazların sayısı ve çeşitliliği artmakta ve bu nedenle saldırı yüzeyi genişlemektedir. Sonuç olarak karşılaştığımız tüm güvenlik açıklarını azaltmak mümkün görünmemektedir. Artık siber saldırıların gerçekleşip gerçekleşmeyeceği mesele değil ne zaman gerçekleşeceği meseledir. Karmaşık saldırıların tespit edilmesi için yeni geliştirilen yaklaşımlardan biri siber tehdit istihbarat paylaşımıdır. Tehdit istihbaratı, tehdidi tanımlayan kanıta dayalı bilgidir ve karar vermeye yardım eder ancak paylaşılmadığı sürece herhangi bir değeri yoktur. Kurumlar gerçek zamanlı olarak kurum içinde elde edilen siber tehdit bilgisini güvenilir paydaşlarla paylaşarak ve dış kaynaklardan elde ettiği tehdit bilgisini kendi güvenlik sistemleri ile entegre ederek hedefli siber saldırılar hakkında durumsal farkındalığı artırabilirler. Ancak, tehdit bilgilerinin tanımlanması ve paylaşılmasında otomasyona izin veren ortak dil, zamanında müdahale için çok önemlidir. Bu amaçla, çeşitli organizasyon ve firma tarafından birçok standart geliştirilmiştir. Bu çalışmada tehdit bilgisinin gösterimi ve paylaşımı için geliştirilen standartlar ve araçlar karşılaştırılmış, ve özel bir blokzincir kullanılarak yeni bir tehdit paylaşım modeli geliştirilmiştir.

Anahtar Kelimeler: siber tehdit istihbaratı, bilgi paylaşımı, siber olay belirtileri, blokzincir

*Dedicated to my family. . .*

# ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitude to my supervisors Aybar Can Acar and Attila Özgit for their continuous support, patience and encouragement throughout my thesis study.

Besides my supervisors, I would like to thank my beloved wife Fatime and my sweet-hearts Berra and Neda for their love, patience and support during the thesis period.

# TABLE OF CONTENTS

ABSTRACT . . . . . iv

ÖZ . . . . . vi

DEDICATION . . . . . viii

ACKNOWLEDGMENTS . . . . . ix

TABLE OF CONTENTS . . . . . x

LIST OF TABLES . . . . . xiv

LIST OF FIGURES . . . . . xv

LIST OF ABBREVIATIONS . . . . . xvii

1 INTRODUCTION . . . . . 1

1.1 Security in Cyber Domain . . . . . 1

1.2 Problem Statement and Goals . . . . . 2

1.3 Vocabulary . . . . . 4

1.4 Thesis Organization . . . . . 5

2 BACKGROUND AND LITERATURE REVIEW . . . . . 7

2.1 Intelligence . . . . . 7

2.2 Cyber Threat Intelligence . . . . . 7

2.2.1 CTI Ecosystem . . . . . 9

2.3 Major Drivers in CTI . . . . . 11

2.3.1 Legislative and Governmental Drivers . . . . . 11

2.3.2 Standardization Efforts . . . . . 12

2.3.3 Commercial Drivers . . . . . 14

2.4 Information Sharing Formats . . . . . 15

2.4.1 Comparison of Threat Reporting Formats . . . . . 16

2.4.2 Low Level Data Formats . . . . . 16

2.4.3 Actionable Data Formats . . . . . 18

2.4.4 Vulnerability and Attack Data Formats . . . . . 19

2.4.5 Threat Reporting Formats . . . . . 21

2.4.6 Other Data Formats . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25
2.5 Transport Protocols . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26
2.5.1 Trusted Automated eXchange of Indicator Information (TAXII) 26
2.5.2 Extensible Messaging and Presence Protocol (XMPP) . . . . . . 27
2.5.3 Other Protocols . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 27
2.6 Data Serialization Techniques . . . . . . . . . . . . . . . . . . . . . . . . . . 27
2.6.1 XML . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 28
2.6.2 JSON . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 28
2.6.3 Other Approaches . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 28
2.7 Data Exchange Models . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29
2.8 Threat Intelligence Tools . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29
2.8.1 Comparison of Threat Intelligence Tools . . . . . . . . . . . . . . . 30
2.8.2 Malware Information Sharing Platform (MISP) . . . . . . . . . . . 34
2.8.3 Collective Intelligence Framework (CIF) . . . . . . . . . . . . . . . . 34
2.8.4 Collaborative Research Into Threats (CRITs) . . . . . . . . . . . . . 35
2.9 Blockchain Technology . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 36
2.9.1 Blockchain . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 36
2.9.2 Permissionless and Permissioned Blockchains . . . . . . . . . . . . 36
2.9.3 Consensus Mechanism . . . . . . . . . . . . . . . . . . . . . . . . . . . . 37
2.9.4 Decision Making . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 38
2.10 Hyperledger Fabric . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 40
2.10.1 Hyperledger Fabric Platform . . . . . . . . . . . . . . . . . . . . . . . 42
2.10.2 Assets . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 43
2.10.3 Chaincode . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 43
2.10.4 Ledger . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 43
2.10.5 Identity . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 44
2.10.6 Membership Service Provider . . . . . . . . . . . . . . . . . . . . . . . 44
2.10.7 Certificate Authorities . . . . . . . . . . . . . . . . . . . . . . . . . . . . 45
2.10.8 Organizations . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 45
2.10.9 Nodes . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 45
2.10.10Consortium . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 47

2.10.11 Channel . . . . . . . . . . 47
2.11 Research . . . . . . . . . . 48
2.12 Challenges . . . . . . . . . . 50
2.12.1 Lack of Interest . . . . . . . . . . 50
2.12.2 Lack of Automation . . . . . . . . . . 50
2.12.3 Legal Uncertainty . . . . . . . . . . 51
2.12.4 Lack of Trust . . . . . . . . . . 51
2.12.5 Safeguarding Sensitive Data and Privacy . . . . . . . . . . 52
2.12.6 Low Quality of Shared Data . . . . . . . . . . 52
2.12.7 Inevitable Costs . . . . . . . . . . 53
2.13 Proposed Solution . . . . . . . . . . 53
2.13.1 Security and Trust . . . . . . . . . . 54
2.13.2 Availability . . . . . . . . . . 54
2.13.3 Accountability . . . . . . . . . . 55
2.13.4 Efficiency . . . . . . . . . . 55
2.13.5 Low cost . . . . . . . . . . 55
2.13.6 Privacy . . . . . . . . . . 55
3 METHODOLOGY . . . . . . . . . . 57
3.1 System Goals and Requirements . . . . . . . . . . 57
3.1.1 Security and Trust . . . . . . . . . . 57
3.1.2 Integrity and Accountability . . . . . . . . . . 57
3.1.3 Availability . . . . . . . . . . 58
3.1.4 Automation . . . . . . . . . . 58
3.1.5 Privacy . . . . . . . . . . 58
3.1.6 Client Application . . . . . . . . . . 58
3.2 Implementation . . . . . . . . . . 59
3.2.1 Environment Setup . . . . . . . . . . 59
3.2.2 System Architecture . . . . . . . . . . 61
3.2.3 Application Scenario . . . . . . . . . . 64
3.2.4 Chaincode . . . . . . . . . . 65
3.2.5 Data Set . . . . . . . . . . 66

4 RESULTS . . . . . . . . . . 67
4.1 Experiment Setup . . . . . . . . . . 67
4.1.1 Metrics . . . . . . . . . . 67
4.1.2 Open Source Testing Tool . . . . . . . . . . 67
4.1.3 Test Configurations . . . . . . . . . . 68
4.1.4 Performance Test Configurations . . . . . . . . . . 68
4.1.5 Functionality Test Configuration . . . . . . . . . . 70
4.2 Discussing Findings . . . . . . . . . . 71
4.2.1 Performance Test Findings . . . . . . . . . . 71
4.2.1.1 Payload Size . . . . . . . . . . 71
4.2.1.2 Client Application Server . . . . . . . . . . 72
4.2.1.3 State Database . . . . . . . . . . 73
4.2.1.4 Block Size . . . . . . . . . . 75
4.2.1.5 Message Count . . . . . . . . . . 77
4.2.1.6 Channel Population . . . . . . . . . . 78
4.2.1.7 Query Operations . . . . . . . . . . 79
4.2.1.8 Comparison . . . . . . . . . . 82
4.2.2 Functionality Test Findings . . . . . . . . . . 84
5 CONCLUSION . . . . . . . . . . 87
5.1 Contribution and Discussion . . . . . . . . . . 87
5.2 Limitations and Future Work . . . . . . . . . . 88
REFERENCES . . . . . . . . . . 89
A CHAINCODE . . . . . . . . . . 101
B SAMPLE EVENT . . . . . . . . . . 103

# LIST OF TABLES

Table 2.1 Comparison of Threat Reporting Formats . . . . . . . . . . . . . . . . . . . . . 16
Table 2.2 Comparison of Cyber Threat Intelligence Tools 1 . . . . . . . . . . . . . . . 31
Table 2.3 Comparison of Cyber Threat Intelligence Tools 2 . . . . . . . . . . . . . . . 32
Table 2.4 Challenges of CTI Sharing . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 54
Table 2.5 Solutions of Proposed Model . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 56

Table 3.1 Environmental Setup for Hyperledger Fabric Network . . . . . . . . . . . 59

Table 4.1 Test Configurations for Performance Evaluation . . . . . . . . . . . . . . . . 69
Table 4.2 Test Results for Different Payload Size . . . . . . . . . . . . . . . . . . . . . . . 73
Table 4.3 Test Results for Different Server Processing Modes . . . . . . . . . . . . . 75
Table 4.4 Test Results for Different Database Options . . . . . . . . . . . . . . . . . . . 76
Table 4.5 Test Results for Different Block Size . . . . . . . . . . . . . . . . . . . . . . . . 78
Table 4.6 Test Results for Different Message Count . . . . . . . . . . . . . . . . . . . . . 80
Table 4.7 Test Results for Different Channel Population . . . . . . . . . . . . . . . . . 80
Table 4.8 Test Results for Query Transactions . . . . . . . . . . . . . . . . . . . . . . . . . 83
Table 4.9 Test Results for Privacy Functionality . . . . . . . . . . . . . . . . . . . . . . . 85
Table 4.10 Test Results for Automated Secure Sharing Functionality . . . . . . . . . 86

# LIST OF FIGURES

Figure 1.1 World's Biggest Data Breaches . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

Figure 2.1 Ratcliffe's 3-I Model . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

Figure 2.2 Cyber Threat Intelligence Ecosystem . . . . . . . . . . . . . . . . . . . . . . . 10

Figure 2.3 Decision Making Flow Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . 39

Figure 2.4 Comparison of Private Blockchain Solutions . . . . . . . . . . . . . . . . . 41

Figure 2.5 Basic Hyperledger Fabric Network . . . . . . . . . . . . . . . . . . . . . . . . . 46

Figure 2.6 Hyperledger Fabric Network With Two Channels . . . . . . . . . . . . . 48

Figure 3.1 System Architecture . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 63

Figure 3.2 User Interface of Client Application . . . . . . . . . . . . . . . . . . . . . . . 64

Figure 4.1 Blockchain Network Architecture for Performance Test . . . . . . . . . 70

Figure 4.2 Blockchain Network Architecture for Functionality Test . . . . . . . . . 71

Figure 4.3 Throughput of Create Transactions for Different Payload Size . . . . 72

Figure 4.4 Latency of Create Transactions for Different Payload Size . . . . . . . 72

Figure 4.5 Throughput of Create Transactions for Different Server Processing Modes . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 74

Figure 4.6 Latency of Create Transactions for Different Server Processing Modes . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 74

Figure 4.7 Throughput of Create Transactions for Different Database Options . 75

Figure 4.8 Latency of Create Transactions for Different Database Options . . . . 76

Figure 4.9 Throughput of Create Transactions for Different Block Size . . . . . . 77

Figure 4.10 Latency of Create Transactions for Different Block Size . . . . . . . . . 78

Figure 4.11 Throughput of Create Transactions for Different Message Count . . . 79

Figure 4.12 Latency of Create Transactions for Different Message Count . . . . . 79

Figure 4.13 Throughput of Create Transactions for Different Channel Population 81

Figure 4.14 Latency of Create Transactions for Different Channel Population . . 81

Figure 4.15 Throughput of Query Transactions . . . . . . . . . . . . . . . . . . . . . . . . . 82

Figure 4.16 Latency of Query Transactions . . . . . . . . . . . . . . . . . . . . . . . . . . . . 82

Figure 4.17 Throughput Results for Comparison . . . . . . . . . . . . . . . . . . . . . . . . 83

Figure 4.18 Throughput Results for Comparison . . . . . . . . . . . . . . . . . . . . . . . . 84

# LIST OF ABBREVIATIONS

| | |
|---|---|
| CAPEC | Common Attack Pattern Enumeration and Classification. |
| CCSS | Configuration Scoring System. |
| CERT | Computer/Cyber Emergency Response Team. |
| CIF | Collective Intelligence Framework. |
| CoA | Course of Action. |
| CPE | Common Platform Enumeration. |
| CRITs | Collaborative Research Into Threats. |
| CVE | Common Vulnerabilities and Exposures. |
| CVSS | Common Vulnerability Scoring System. |
| CWE | Common Weakness Enumeration. |
| CWSS | Common Weakness Scoring System. |
| CybOX | Cyber Observable Expression. |
| IDS | Intrusion Detection System. |
| IOC | Indicator of Compromise. |
| IODEF | Incident Object Description Exchange Format. |
| IPS | Intrusion Prevention System. |
| IPFIX | Internet Protocol Flow Information Export. |
| MAEC | Malware Attribute Enumeration and Characterization. |
| MISP | Malware Information Sharing Platform. |
| RID | Real-time Inter-network Defense. |
| SIEM | Security Incident Event Management. |
| STIX | Structured Threat Information Expression. |
| SWID | Software Identification Tags. |
| TAXII | Trusted Automated eXchange of Indicator Information. |
| TTPs | Tactics, Techniques and Procedures. |
| OpenIOC | Open Indicator of Compromise. |
| XCCDF | Extensible Configuration Checklist Description Format. |
| XMPP | Extensible Messaging and Presence Protocol. |

| | |
|---|---|
| YARA | YARA: Another Recursive Acronym, or Yet Another Ridiculous Acronym. |

# CHAPTER 1

# INTRODUCTION

In this chapter motivation behind this study is introduced and research problem is defined.

## 1.1 Security in Cyber Domain

Computerized world has become an essential part of human ever since continuous technological innovations dominated over every aspect of life. There are about 4,4 billion Internet users, which exceeds 50% of the earth population [1]. Besides human beings, governmental and industrial institutions rely on ubiquitous information technology (IT) which is a major indicator and contributor of economic development.

However, ceaseless advancements yield much more complex digital systems (hardware, software and networks) and complexity cause insecurities to raise, as Schneier stated [2]. Besides that, complex systems with various beneficial features boosts the demand and dependency in recent years. As a result of much more dependency on cyberspace; not only the number of vulnerabilities, but also the potential costs and impacts of a successful attack increased.

Besides that, the exponentially developing technology can provide the technical means to identify, isolate and eliminate the risk factors. Therefore, the importance of cyber security is being realized and significant amount of budget and personnel is being reserved both by governments and companies. While victims are increasing cyber security awareness and defenders are enhancing countermeasures, threats evolve rapidly and changing methods in an unprecedented way. Furthermore, threats has not merely changed their tactics, techniques, and procedures (TTPs), they have changed their targets from end users to highly profitable enterprises and government institutions. With the motivation of newly discovered zero-day or unpatched vulnerabilities, and destructive impacts, well-organized threat actors are willing to use weapons range from malware with sandbox evasion techniques to highly persuasive social engineering methods to bypass well-known countermeasures. WannaCry [3] and NotPetya [4] are just two examples of rapidly spreading indiscriminate ransomware attacks which make the landscape ever more dangerous and evolving. According to World Economic Forum survey cyber attacks are in the top ten risks in terms of impact and likelihood [5]. In regard to data breaches, Figure 1.1 shows world's biggest ones involving more than 30,000 stolen records.

While skillful threat actors are choosing diverse attack targets, increasing the speed and sophistication of attacks [7], separated efforts to combat them are inadequate [8]. As a result, new paradigms; 'Cyber Threat Intelligence (CTI)' and 'Information Sharing' needed to response complex threats. Cyber security guards should obtain the up-to-date information about threats and incidents so that they can take security measures proactively to reduce the impacts. Gathering timely threat information and derive intelligence from that info is crucial for defending against contemporary cyber attacks. Additionally, intelligence has no value if it is not disseminated [9]. Information sharing enriches the quality of intelligence and supports the overall security.

## 1.2 Problem Statement and Goals

Information is power in this century, without power one has no ability to stop another [10] and intelligence is the distilled information. Considering the cyber threats—evolving, borderless, asymmetric—as incurable problem, intelligence sharing is invaluable to gain insight about and deter the adversary. However, there are various challenges in this realm:

- First concern is the proliferation of threat sharing standards and tools which

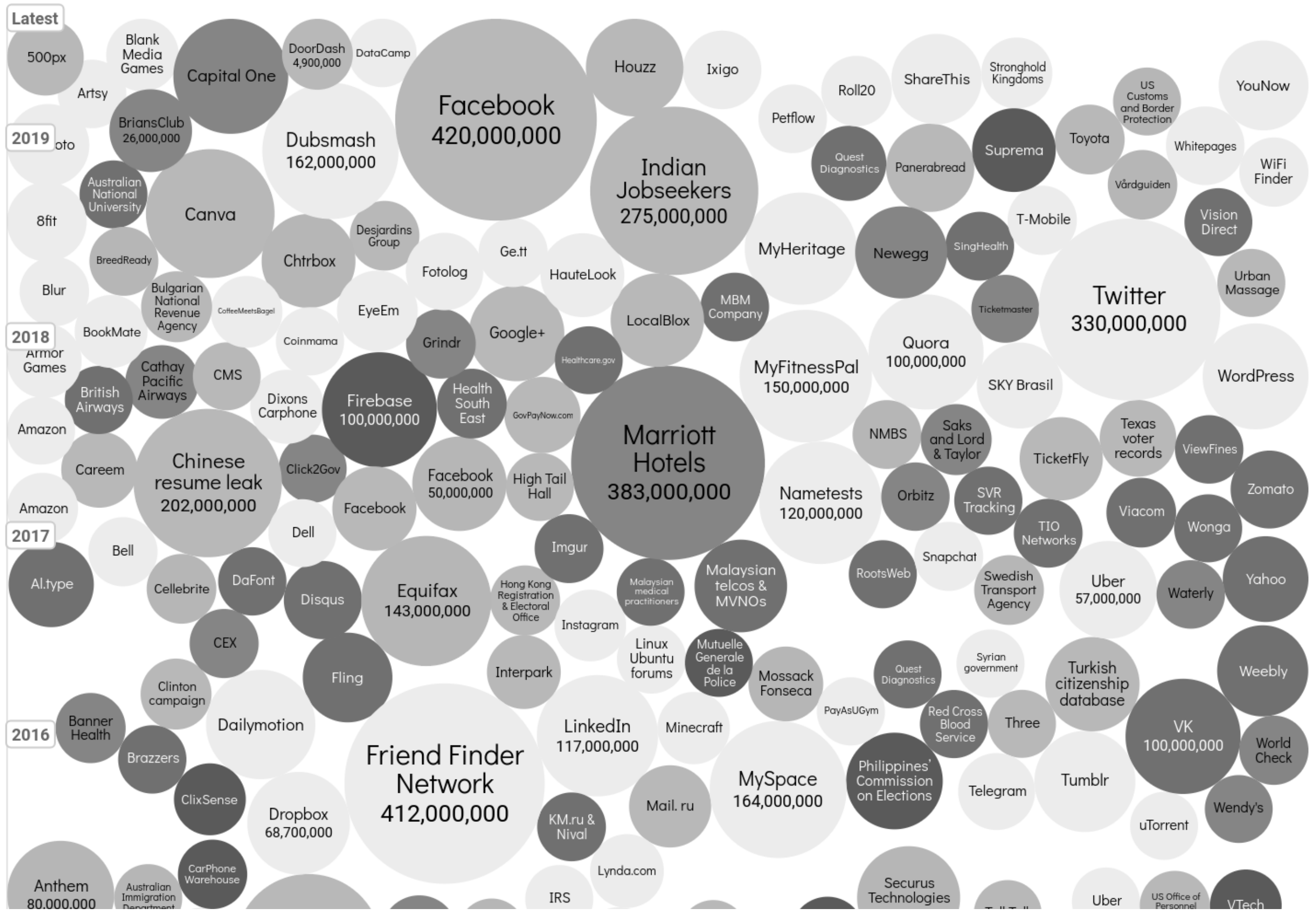


Figure 1.1: World's Biggest Data Breaches

Retrieved from [6]

have different characteristics. Each one offers a various kind of intelligence and provide security professionals with diverse levels of benefits. We need to have basic knowledge about intelligence, key concepts and developed technologies in order to decide the most appropriate one to our use.

- Another concern to deal with is the stakeholders do not want to share cyber threat information because of several hindrances (See Section 2.12).

However, available technologies do not refer to become a remedial to those challenges. Thus, we should consider a new sharing model based on permissioned blockchain technology that handles these issues.

The goals of this study are

- to compare the developed threat information sharing technologies and choose appropriate standard for our use case,

- to propose a new model based on a permissioned blockchain technology.

- to test the functionality, measure the performance and validate the benefits of the proposed model.

## 1.3 Vocabulary

Different namings are present in the cyber threat intelligence domain. ‘Threat intelligence’ or ‘threat information’ expressions are frequently used for CTI. Furthermore, tools that produce, consume or facilitate sharing threat information are named as ‘threat information sharing tools’ or ‘threat intelligence platforms’, and threat sharing standards are called as ‘threat information languages’ or ‘threat reporting formats’ Thus, we also used these terms interchangeably in this study.

## 1.4 Thesis Organization

The rest of this thesis is organized as follows. In Chapter 2, we present a detailed description of CTI, related efforts, data representation approaches, available CTI technologies, basic blockchain concepts and Hyperledger Fabric, and researches and challenges in the field of cyber security information sharing. Chapter 3 describes the methodology that we followed in this study. Chapter 4 presents the evaluation and implementation results. Chapter 5 finally presents the conclusion and future research that we plan to conduct as an extension to this thesis.

# CHAPTER 2

# BACKGROUND AND LITERATURE REVIEW

This chapter presents the comprehensive overview of the threat intelligence domain and related technologies. It also gives brief information about blockchain and Hyperledger Fabric. Lastly, researches, challenges and provided solution are introduced.

## 2.1 Intelligence

As a general definition, Merriam-Webster [11] states intelligence as information related to an enemy.

More detailed ones express that unrefined raw information is obtained, categorized, analyzed, interpreted, and protected in a methodical and purposeful way to produce intelligence and harvest is disseminated to decision-makers for changing the outcomes of the forthcoming situations[12, 13, 14]. Therefore, it is clear that when intelligence supports the decision-making process and enable reasonable insight into forthcoming situations, it reaches its peak value.

## 2.2 Cyber Threat Intelligence

Cyber threat intelligence, also called threat intelligence, is defined in various ways so far.

McMillan stated that threat intelligence is evidence based knowledge about current or rising threats that can be useful to make decisions [15]. Another definition points out threat intelligence as the analysis of an adversary's intent and capacity to damage [16]. Generally, cyber threat intelligence gives information about adversaries and their behaviors that can inform defensive actions. However it is a fuzzy term to understand.

To understand how cyber threat intelligence works, we adapted Ratcliffe's conceptual three-i (interpret, influence, impact) model in Figure 2.1. While the original one is developed for reducing crime, adapted model aim to reduce the risk of the cyber attack. In summary:

- The cybersecurity analysts actively interprets the threat environment in order to determine who the major players are, and what are the significant and emerging threats.
- With information gathered analysts are supposed to influence the thinking of decision makers.
- For risk reduction, decision-makers must bring about an impact on threat landscape and they must direct resources effectively.

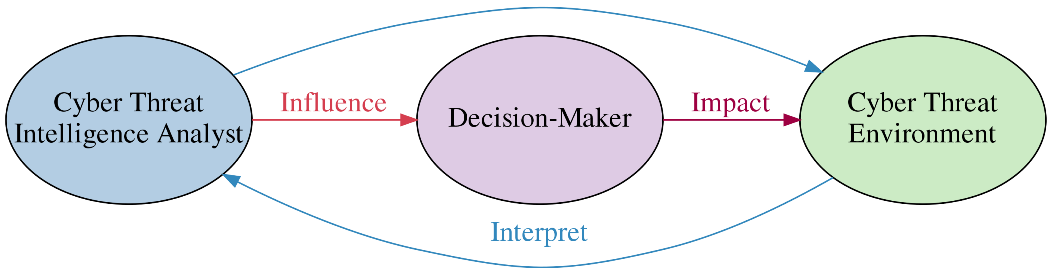


Figure 2.1: Ratcliffe's 3-I Model
Adapted to Cyber Threat Intelligence from [17]

CTI can be categorized into four subdivisions conforming with four levels of intelligence analysis [18]; strategic, operational, tactical and technical. **Strategic CTI** is high-level information — attack trends, future predictions, financial impacts of attacks — used by executive boards. **Operational CTI** is information about imminent attacks targeted to the organization and consumed by cyber security managers [19]. **Tactical and Technical CTI** is our main focus and discussed in Section 2.2.1.

### 2.2.1 CTI Ecosystem

In order to understand general concept of CTI we should examine the CTI ecosystem in Figure 2.2 step by step.

- Threat data can be obtained from various sources like commercial data providers, open/private sharing communities, internal sources, security tools etc.
- If collected data is low level (unstructured) it is analyzed and refined by security analysts. This analysis yield to tactical and/or technical CTI.
- If received data have been distilled in advance it can be used as threat intelligence.
  - **Tactical CTI** is information about how adversaries attack, often referred to as Tactics, Techniques, and Procedures (TTPs) and tools of threat actors [19]. With tactical TI, cybersecurity practitioners ensure that their defense policy is sufficient and up-to-date. White papers and communicating with partners provide insight about the attackers' trending methods and general approaches.
  - **Technical CTI** is basic indicators of an attack e.g., IP/URL/Domain addresses, malware hashes, dll names, registry keys, email attachments, links etc. They are utilized by technical means [19] such as Security Incident Event Monitoring devices (SIEMs), intrusion detection and prevention systems (IDS/IPSes), firewalls. These devices detect the intrusions, block the connection attempts to suspected addresses by consuming technical threat intelligence.
- Furthermore, if intelligence data need to be restructuring, it should be redefined in machine-readable format to use in our platform and/or to share with other peers. Threat reporting formats are employed for structuring purpose.
- Formatted threat intelligence now is ready for the consumption of the community and warn for the emerging threats in a timely-basis. Threat sharing platforms share data to other connected peers in a bidirectional way.

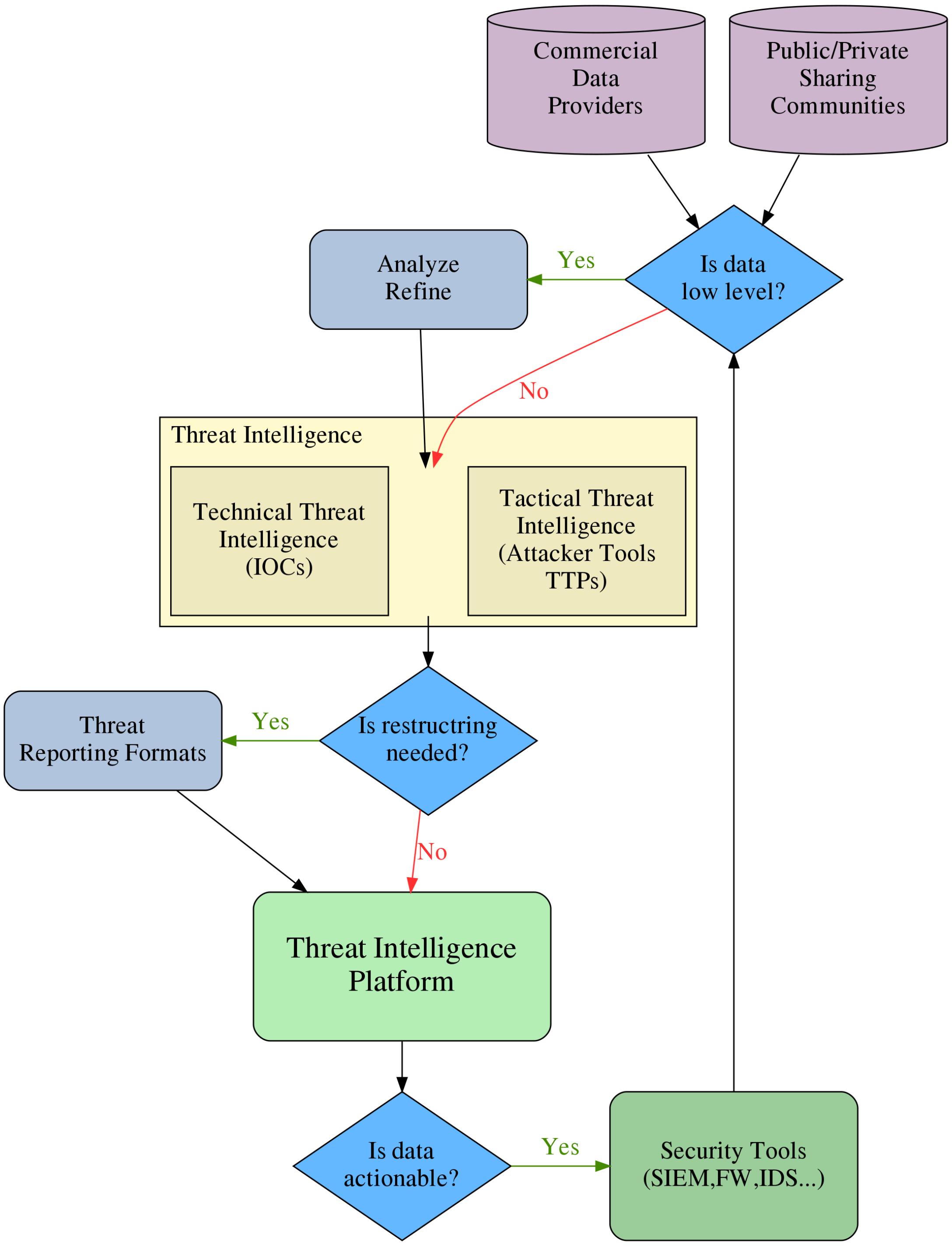

Commercial Data Providers
Public/Private Sharing Communities
Is data low level?
Yes
Analyze Refine
No
Threat Intelligence
Technical Threat Intelligence (IOCs)
Tactical Threat Intelligence (Attacker Tools TTPs)
Is restructring needed?
Yes
Threat Reporting Formats
No
Threat Intelligence Platform
Is data actionable?
Yes
Security Tools (SIEM,FW,IDS...)

Figure 2.2: Cyber Threat Intelligence Ecosystem

- Based on the ability of the standard, it is also possible to define actionable items in these reporting formats. Therefore, actionable decisions can be employed by security tools.

## 2.3 Major Drivers in CTI

It is more painful than before to detect and analyze a cyber incident since cyber attacks become more complicated. Adversaries moving fast and stealthy. To keep pace and combat this new age of attacks, nations embrace and emphasize the information sharing, some of them introduce encouraging laws. Besides that, security companies and organizations integrate CTI technology with their products and/or provide intelligence feeds, and cyber emergency response teams (CERTs) throughout the world increasingly use automation to produce, consume and dissect threat information.

### 2.3.1 Legislative and Governmental Drivers

NATO Cooperative Cyber Defence Center of Excellence (CCDCOE) is established after the Estonian cyber crisis [20] in 2008. This establishment's major role is improving the cooperation and information sharing among allies [21]. Center is also working on the applicability of international law in the cyber field.

In Germany and European Union, information sharing is guaranteed by law for critical infrastructures [22, 23]. NATO Computer Incident Response Capability (NCIRC) and CERT-EU signed an arrangement that allows threat information sharing in 2016 [24]. EU cyber response team is able to share and access threat data provided through Malware Information Sharing Platform (MISP) which is developed by NATO, Belgian Armed Forces and Luxembourg Computer Incident Response Center (CIRCL). Detailed information about MISP is presented in Section 2.8.

In the U.S., two important legislation have come into effect in 2015. President issued an executive order directing the Department of Homeland Security (DHS) to bolster the constitution of Information Sharing and Analysis Organizations (ISAOs) [25].

Additionally, to enhance sharing threat information and defensive measures between public and private partners Cybersecurity Information Sharing Act is approved [26]. In 2017, U.S. National Cybersecurity and Communications Integration Center (NC-CIC), 2009 established organization, changed its organizational structure to integrate similar functions of US-CERT and Industrial Control Systems Cyber Emergency Response Team (ISC-CERT) in one umbrella to enhance the cyber security efficiency [27]. NCCIC receives incident reports from government institutions, sectoral CERTs and private partners and convert these reports to actionable alerts. Likewise, Cyber Threat Intelligence Integration Center (CTIIC) is established in 2015 to analyze foreign cyber threats [28].

Turkish National Cyber Incident Response Center also established CERT communication platform in 2017 to share threat information and countermeasures between trusted partners [29]. There are news that Turkey prepared a draft cyber security law but it has not finished yet [30]. However, there are some legislative acts and regulations regarding information technology.

### 2.3.2 Standardization Efforts

To keep up the speed of the adversaries, automation and common language is critical. Thus, several standardization endeavors have seen throughout the development of cooperation between cyber security centers worldwide.

Some of these standards convey general information about incident related data but some of them captures the intelligence information in a machine readable form so that it can be shared among different platforms. These standards are called cyber threat information languages or threat reporting formats which specify how data is stored. Besides a standard language, an exchange protocol which specifies how data is transmitted is necessary for threat sharing. We will mention these standard languages and exchange protocols in this section.

U.S. not-for-profit organization MITRE has developed a set of standards and languages that enables general data sharing standards which related with threats implicitly like Common Vulnerabilities and Exposures (CVE), Common Platform Enumeration (CPE), Common Attack Pattern Enumeration and Classification (CAPEC). These developments are based on a purpose; making security manageable and measurable [31]. Furthermore, for more specific aim, exchanging explicit threat data, Structured Threat Information eXpression (STIX) standard language and Trusted Automated Exchange of Intelligence Information (TAXII) exchange protocol has developed in 2012 [32, 33]. The DHS Office of Cybersecurity and Communications funded MITRE for development.

MITRE manages U.S. federal research and development centers. By the way, according to Stoll's influential cyber espionage book [34], first published in 1989 but still continue to be relevant regarding cyber security and threat sharing issues, MITRE was not only a defense contractor but also engaged with the Central Intelligence Agency (CIA) and National Security Agency (NSA).

Some of the information sharing standards (STIX, TAXII, etc.) that had developed by MITRE is now being developed by nonprofit consortium, Organization for the Advancement of Structured Information Standards Cyber Threat Intelligence Technical Committee (OASIS CTI TC) since 2015 [35].

Internet Engineering Task Force (IETF) is also contributing these efforts with several standards. IETF Managed Incident Lightweight Exchange Working Group (MILE WG) has developed Incident Object Description Format (IODEF) standard in 2007 [36] for expressing incident data and Real-time Inter-network Defense (RID) in 2010 [37] as an exchange protocol. It was an alternative to TAXII but no longer maintained. At the moment IETF is focusing on Extensible Messaging and Presence Protocol (XMPP) for a sharing data [38].

Cyber security firms are also developing proprietary threat sharing standard languages like OpenIOC of Mandiant, FireEye [39]. But these standards usually have limited use in other platforms.

In practice, standards and protocols are used with additional tools such as format converters, tool plugins, and APIs for adoption of cyber threat sharing. So availability of these extra tools enrich the standard and protocol efficiency and usability.

These mentioned standards are explained in Section 2.4 briefly.

### 2.3.3 Commercial Drivers

Cyber Security companies also understand the importance of threat intelligence sharing. They are acquiring other threat intelligence firms, releasing new threat sharing platforms or initiating services for subscribers.

Firstly, AlienVault Open Threat Exchange (OTX) is released in 2012 [40]. Check Point released threat sharing platform ThreatCloud IntelliStore in May 2014 [41]. Facebook launched ThreatExchange, a sharing platform about cyber threats in February 2015 [42], IBM released cloud-based one X-Force Exchange in April 2015 [43]. Symantec launched a threat intelligence service DeepSight in November 2015 [44], FireEye bought CTI firm iSight Partners [45] in January 2016, NC4 acquired Soltra Edge from finance sector [46] in November 2016. Lastly, AT&T acquires another threat intelligence company AlienVault [47]. These are the signs of the progress in the area.

In summary cyber threat intelligence technologies can be classified into three categories:

- Information sharing formats which aim to represent data, they are categorized in Section 2.4,
- Protocols for transmission of threat data described in Section 2.5
- Data serialization techniques expressed in Section 2.6,
- Data exchange models depicted in Section 2.7,
- Threat intelligence sharing platforms summarized in Section 2.8.

Following section starts with the data representation formats.

## 2.4 Information Sharing Formats

The focus of this section is on the standard data formats that are used for cyber threat intelligence sharing.

Information security practitioners use different types of data while encountering incidents. Common language is very important for incident handling process to agree on the same understanding with the other incident responders. As a result, broadly adopted sharing formats which are agreed upon by community prevent the misinterpretation of disseminated data. Furthermore, the other advantage of adopting standard formats is broad support from existing security tools [48].

Information sharing formats are developed by several organizations thus far. There is not a one and only solution that fits for all circumstances and/or data. Even though some of them have distinct characteristics, some standards overlaps with each other. For that reason, overall comparison is needed to choose the appropriate solution for different sharing scenarios.

Information sharing formats can be classified according to data of which they illustrate. These are;

- Low level raw data (network traffic), see Section 2.4.2.
- Basic incident indicators, also known as actionable info (artifacts; file hash, IP/Domain address, registry key information, etc.), see Section 2.4.3.
- Vulnerability/weakness and attack pattern information (vulnerability ID, vulnerability score), see Section 2.4.4.
- Incident/Threat reporting data, see Section 2.4.5.

Formats that express the incident/threat data can make use of other format types to give the most through information about incidents. Thus, our comparison focuses on threat reporting data formats.

### 2.4.1 Comparison of Threat Reporting Formats

We evaluate most relevant threat reporting formats according to their support of other information sharing formats. The summary of our comparison is on Table 2.1.

Table 2.1: Comparison of Threat Reporting Formats

| Purpose | Standard Formats | Threat Reporting Formats | | |
|---|---|---|---|---|
| | | STIX | IODEF | MISP |
| Low-level Data | Pcap | + | − | − |
| | NetFlow, IPFIX | + | − | + |
| | CEF | + | − | + |
| Actionable Data | IDS rule | + | − | + |
| | YARA rule | + | − | + |
| | OpenIOC | + | − | + |
| | CybOX | + | − | + |
| | MAEC | + | + | − |
| | MMDEF | + | + | − |
| Vulnerability and Attack Data | CVE | + | + | + |
| | CWE | + | + | − |
| | CPE | + | + | + |
| | CVSS | + | + | + |
| | CWSS | − | + | − |
| | CAPEC | + | + | − |

Details about the standard formats are explained in the following sections with aforementioned classification.

### 2.4.2 Low Level Data Formats

The formats in this category; Pcap, NetFlow, IPFIX and CEF, represents network level data collected by security monitoring tools.

### Pcap

Pcap is the format to store or transmit captured network traffic and used by many security tools [49]. It is open source and developed by Tcpdump group. Tcpdump [49], Wireshark [50], Snort [51], NetworkMiner [52] and libpcap [53] are a few of the tools and libraries that can parse and analyze pcap data.

### NetFlow

NetFlow is the protocol for exporting traffic statistics as IP flow records from NetFlow enabled network devices to analyze the network traffic by its source, destination and volume [54]. SiLK [55], Argus [56] and Ntopng [57] are examples of NetFlow tools/analyzers.

### IP Flow Information Export (IPFIX)

IPFIX is a derivative of NetFlow and became an IETF standard [58]. It is used in SiLK, Argus and libIPFIX [59].

### Common Event Format (CEF)

CEF is a syslog based event format used in SIEMs to transmit event information between event producers and consumers [60]. It is initially developed by ArcSight and used by other vendors. It enhances the interoperability of security-related information from different devices. It contains a syslog prefix, header and extension and these sections describe the event with device info, event type, event description, network addresses, users, files, firewall rules and other OS artifacts.

### 2.4.3 Actionable Data Formats

Actionable data formats define the attributes of threats and used for incident detection. They are usually referred as indicators of compromise (IOCs). IDS rules, YARA rules, OpenIOC, CybOX, MAEC and MMDEF and are examples of these formats.

#### Intrusion Detection System (IDS) Rules

IDS rules are used by Network Intrusion Detection/Prevention Systems (NIDS/NIPS) to analyze and detect the malicious activities in the network traffic. Popular open source rule-based IDS/IPSes are Snort [51], Suricata [61] and Zeek (formerly Bro) [62].

#### YARA rules

YARA rules are used to dissect, identify and classify malware samples within YARA tool [63]. YARA rules based on strings (regular expressions, hexadecimal and text strings) and boolean expressions. Security analyst can create rules and hunt for certain attributes of a malware. YARA, the pattern matching swiss army knife for every security investigator, reduces the costly process of reverse engineering. It is originally developed by Victor Manuel Alvarez and open source. There are many security tools using this format—VirusTotal, FireEye, TrendMicro, CrowdStrike, Cuckoo Sandbox and ThreatConnect to mention but a few [64].

#### OpenIOC

OpenIOC is an XML-based language developed for recording IOCs and basically describes indicators, malware artifacts and TTPs [65]. It also allows to embed an IDS (Snort) or YARA signature (rule) into an indicator item. To create and edit OpenIOC data, ioc-writer was developed [66]. OpenIOC schema is initially created and used by Mandiant, an incident response company which is acquired by FireEye, and open sourced later.

### Cyber Observable eXpression (CybOX)

CybOX is a XML-based structured language for representing cyber observables; in other words IOCs [67]. An observable of an event such as a file, a network packet or a registry key can be represented as a CybOX object. CybOX is also used in CAPEC and MAEC standards. There is a python library for parsing, changing and creating CybOX data [68].

### Malware Attribute Enumeration and Characterization (MAEC)

MAEC is a structured language for sharing malware behavior, abilities and attributes to reduce the analysis efforts [69]. In the graph based MAEC data model, the MAEC relationships associates the MAEC objects with each other. A relationship defines how objects are related. Behavior, collection, malware action, malware family and malware instance are top level objects in the data model. There are programs that convert the output of malware analysis tools (Cuckoo, Anubis, GFI and VirusTotal) into MAEC format. However only VirusTotal conversion tool is compatible with the new JSON format used in MAEC version 5.

### Malware Metadata Exchange Format (MMDEF)

MMDEF is an XML schema to exchange malware samples, metadata and behavioral information [70]. Antivirus industry is using MMDEF to aid malware information exchange. MMDEF can be used in XML version of MAEC to describe malware. Cuckoo sandbox was using MMDEF in its older versions.

## 2.4.4 Vulnerability and Attack Data Formats

Vulnerability and attack data formats define the meaning of a vocabulary utilized by the IT community. These formats enable common understanding about vulnerabilities, weaknesses, attack patterns and quantitative measurement of threats. For quantitative measurement of threats scoring formats are utilized. CVE, CWE, CPE, CVSS, CWSS and CAPEC are the formats most commonly used in this category.

**Common Vulnerabilities and Exposures (CVE)**

CVE is a list of known security vulnerabilities and exposures [71]. Each entry in the list contains identification number, description and at least one public reference. CVE list is maintained by MITRE and feeds the U.S. National Vulnerability Database (NVD). CVE data is available in text, CSV, HTML and XML format.

**Common Weakness Enumeration (CWE)**

CWE is a community-developed list of software weaknesses that can appear in architecture, design, code and implementation [72]. These weaknesses can lead to security vulnerabilities. The main objective of the project is to stop vulnerabilities at the source by raising awareness among software developers and acquirers. Although it resembles to CVE, it does not define a specific vulnerability for a particular software. It describes a weakness in multiple software types. CWE list is available in CSV, XML and HTML format.

**Common Platform Enumeration (CPE)**

CPE is a structured naming scheme for software applications, operating systems and hardware appliances [73]. It is based on the generic syntax for Uniform Resource Identifiers (URIs). CPE defines versions of the products but does not define specific installations and serial numbers. Main goal of the standard is to facilitate the inventory of IT products. MITRE handed over the development to NIST by whom a searchable CPE dictionary is currently maintained.

#### Common Vulnerability Scoring System (CVSS)

CVSS calculates criticalness and severity of a vulnerability in software or hardware, and it also determines the consequences of a successful exploit of that vulnerability with associated CVE ID [74]. It describes the key characteristics — attack vector, complexity, required privileges, user interaction, scope — and effects — impact to confidentiality, integrity and availability — of a vulnerability and calculates a score based on defined metrics. It is developed by FIRST with support from industry, government and academia.

#### Common Weakness Scoring System (CWSS)

CWSS measures weaknesses similar to CVSS but does not calculate a specific weakness in an individual software [75]. It can rate and prioritize a weakness for different software applications based on their associated CWE IDs. It is developed by MITRE, last version is released at September 2014 and further development is unclear as stated by authors.

#### Common Attack Pattern Enumeration and Classification (CAPEC)

CAPEC is an inventory list of attack patterns which are classified hierarchically based on mechanisms and target domains [76]. Attack steps, prerequisites, vulnerabilities, solutions and related techniques are described in each item. CAPEC helps to understand attacks and allows security practitioners to view the vulnerabilities from the attacker's perspective. CAPEC data is available in CSV and XML format.

### 2.4.5 Threat Reporting Formats

Threat Reporting formats describe threats and incidents comprehensively in a structured fashion and combine multiple types of information like vulnerability, indicator, attack method, remediation procedure. They are mostly based on XML or JSON schemes and these schemes are explained in 2.6. The most relevant and up-to-date threat reporting formats are STIX and IODEF. They provide the most extensive data about threats and used in threat intelligence tools. Besides it is mainly a threat sharing platform, Malware Information Sharing Platform (MISP) has its own JSON-based data representation format. Therefore it is worth to consider it as a threat reporting format.

### Structured Threat Information eXpression (STIX)

STIX is an open source, structured, extensible and modular language to exchange cyber threat intelligence [32, 77, 78, 79]. It has two major versions STIX 1 and STIX 2 that the main difference is the serialization method. STIX 1 is XML-based while STIX 2 employs JSON scheme. Its government driven development have been initiated by MITRE and handed over to OASIS in June 2015.

STIX 1 represents indicators, incidents, targets, attack campaigns, threat actors, attacker tactics and courses of action (CoA) in its data model. Although it is very complex to implement [79], it is asserted as a de-facto standard for expressing the threat information [80] and mostly adopted standardized format [81]. STIX 1 employs CybOX for representing low-level indicator data such as a file hash, a process, a registry key. It can be used for threat analysis, indicator definition, incident response management and intelligence sharing. STIX 1 uses IDS and YARA signatures, CybOX, OpenIOC, MAEC, MMDEF, CVE, CPE, CAPEC, CWE, TLP, CVSS formats. Pcap, NetFlow and CEF formats can also be used via CybOX. Malware artifacts, Command and Control (C&C) activities, IP addresses, domains, URLs, malicious activities, e-mail messages, files, compromised credentials are several examples of incident data that can be expressed. Because of its complex structure it is mostly used in enterprise organizations. Python based tools exist for parsing, manipulating and generating STIX 1 data, and converting OpenIOC indicators to STIX 1 indicators [82].

In STIX 2 more lightweight and simple JSON scheme is adopted [78]. In STIX 1 an IOC can appear as a STIX Indicator and as a CybOX Observable which is very confusing. CybOX objects are now represented as STIX Cyber Observable to resolve the complication. Additionally, all STIX Domain Objects (SDOs) are top level objects and object relationships are expressed by STIX Relationship Objects(SROs). It resembles a graph with nodes as SDOs and edges as SROs. In favor of enhancing the confidence of threat data Sightings object is established.

From the CTI tools viewpoint; MISP has import and export capability of STIX data [83], CIF and CRITs have the ability to parse STIX data with an extra modules [84].

#### *Support for standard data formats*

Pcap and NetFlow data can be represented in STIX version 1 via CybOX object. STIX version 2 patterning feature allows to use NetFlow records. ArcSight has a STIX Python client that convert STIX version 1 XML format to CEF format.

IDS and YARA rules can be used in STIX version 1 indicator type. STIX version 2 claims to provide integration with IDS and YARA rules with patterning component but a converter is not developed yet. It is possible to convert OpenIOC data to STIX version 1 indicator type [85]. CybOX language has been integrated into STIX 2 today and STIX Cyber Observables are proposed instead of CybOX objects [86]. As a result, the CybOX language will not be supported in the future. MAEC version 5, unlike previous XML versions, uses JSON serialization and compatible with STIX version 2. STIX version 1 have also support for this MMDEF standard.

CVE, CPE and CAPEC are used in both versions of STIX. CWE and CVSS formats are only adopted in STIX version 1 scheme.

For comparison see Table 2.1.

### Incident Object Description Format (IODEF)

IODEF is a structured schema to express incidents based on XML-based information model [36]. Data model allow to encode incident related info such as hosts, networks, attack methods, vulnerabilities. IODEF does not have support for low level formats and indicators. Extensions for indicators has been added to IODEF data model with the IODEF 2 [87] [88]. Its development is being conducted by IETF Managed Incident Lightweight Exchange (MILE) Working Group [89]. According to their web page, MILE WG is working on JSON bindings of IODEF and XMPP for information exchange format.

According to [81], 23% of threat intelligence sharing companies use IODEF. Furthermore, CIF, one of the information sharing platforms, can utilize this format [80].

#### *Support for standard data formats*

IODEF supports MMDEF and XML version of MAEC. It also adopts CWE, CPE, CVSS, CWSS and CAPEC in its scheme.

For comparison see Table 2.1.

### Malware Information Sharing Platform (MISP)

MISP, supported by Computer Incident Response Center Luxembourg (CIRCL) and European Union, is a threat sharing platform that is used to share, store and correlate threat data [90]. Nevertheless, it is not merely a software but also has its own data model to express incidents. In its JSON-based format, objects are used to describe incidents and relationships to link objects. MISP can also export IDS, OpenIOC, STIX format.

For the sake of its character as a threat sharing tool, MISP is also defined under Section 2.8.

#### *Support for standard data formats*

MISP has NetFlow object definition and a module to export CEF format.

Furthermore, it allows to export data as Snort, Suricata, Bro IDS rule and has capability to import and export OpenIOC format. It has also an object to describe YARA rules. There is convert to STIX 1 and import STIX 1 options in MISP thereby CybOX data is supported.

MISP has no support for MAEC format but it is planned to develop a MISP module for the import and export of the MAEC format.

MISP also uses the CVE and CPE data. MISP uses CVE IDs in its vulnerability object and has an expansion module for obtaining CVE data.

For comparison see Table 2.1.

### 2.4.6 Other Data Formats

Following formats are used for different purposes. Thus, we do not put into in those categories mentioned above.

#### Traffic Light Protocol (TLP)

TLP protocol labels the sensitive information to ensure that only appropriate viewers can access to it and express the data sharing boundaries [91]. It utilize four colors (white, green, amber, red) to express sharing levels of data. Recipient must get clear consent from original source to share the information more broadly. TLP protocol is used in sharing sensitive threat data by security organizations. STIX, MISP, CIF, CRITS are using TLP protocol.

#### Extensible Configuration Checklist Description Format (XCCDF)

XCCDF is an XML-based language that represents security policy rules for specific target systems [92]. For example, a policy for maximum period of time to expire users' passwords and force a change. It is developed by NIST and became an ISO standard. It can be used in IODEF.

#### Common Configuration Scoring System (CCSS)

CCSS weighs the impact of misconfiguration of a specific asset with scoring metrics like CVSS standard [93]. It is based on CVSS and aids to observe the comprehensive security posture. It is developed by NIST and can be used in IODEF.

#### Software Identification (SWID) Tags

SWID Tags provide XML-based definitive labeling information for software assets [94]. Unlike CPE, SWID tags are not provided by volunteers but product owners. Furthermore, tags record serial numbers for specific installations. The structure of a tag specified in ISO standard, and includes name, edition, version and serial number. Tagging helps to automate the software inventory for asset management.

## 2.5 Transport Protocols

Transport protocols used as information exchange mechanisms. TAXII and XMPP are most notable ones that used for threat intelligence sharing. RESTful interfaces, Email and Twitter are the other options we mentioned. The security of the underlying transport protocol determines the security of the data sharing process [48]. Thus, it is important to find proper solutions for this problem in this layer.

### 2.5.1 Trusted Automated eXchange of Indicator Information (TAXII)

TAXII is an application layer protocol that enables threat information sharing among trusted parties [95]. It exchanges threat information over HTTPS and particularly designed for STIX formatted data. Its government driven development have been initiated by MITRE and handed over to OASIS in June 2015.

### 2.5.2 Extensible Messaging and Presence Protocol (XMPP)

XMPP is an open standard for real-time messaging and presence information [96]. It is designed to be extensible and widely adopted by emerging technologies.

XMPP-Grid, firstly published in [97], used for exchanging threat information. An IETF driven effort is also ongoing [38]. It is named as pxGrid in Cisco commercial products and used for integrating security products.

### 2.5.3 Other Protocols

There are some other protocols that are employed for threat data exchange. RESTful interfaces are gaining popularity because of their simplicity. They are request-response based and stateless. They provide to create, read, update and delete resources. Resources are identified by the URIs. REST is commonly implemented with HTTP protocol and can be secured with HTTP-based services. Furthermore, interface access can be allowed through an API key.

Email can be used for exchange of security related data in different formats. Even though it is widely used by national CERTs, it is not a completely reliable solution because scanned emails can be blocked by email filters along the way.

Furthermore, Twitter is another communication channel used for threat information exchange. Twitter accounts are public by default but can also be private which are only accessible by users who are approved by account holder. It is a considerable tool for exchanging short messages and alerts.

## 2.6 Data Serialization Techniques

Data should be translated into stream of bytes with serialization in order to be stored and shared between different systems. There are different serialization methods. Choice of a method affect volumes of data, processing performance, complexity of relations between objects and available tools that works with chosen method [48].

XML and JSON are commonly used methods for threat data. By the way, nowadays popular formats presented in Section 2.4.5 choosing JSON over XML because of its simplicity. We also mentioned text, CSV and raw log formats in the following sections.

### 2.6.1 XML

Extensible Markup Language (XML) is a standard for encoding data [98]. It is both human and machine readable. It is extensible because its tags are not predefined and custom tags can be defined for any arbitrary data structure. XML encoded document is structured hierarchically and based on nested elements. XML-based information sharing standards define their vocabulary with different element types. XML documents can be easily parseable but have verbosity in representation.

### 2.6.2 JSON

Javascript Object Notation (JSON) is an encoding standard based on collections of name-value pairs [99]. JSON documents are also both human readable and machine parseable. Like XML, it has a generic and extensible data description syntax with which any information can be represented. As compared to XML, it is more lightweight, simple, adequate for threat data and preferred one by developers [86].

### 2.6.3 Other Approaches

There are other text formats which are used for threat data sharing. In high-level reporting, security information is represented in text format. It is widely used in emails and security advisories. Its main usage is human to human communication so machine processing is very difficult.

Comma-separated values (CSV) is a text format where field values are separated by an arbitrary character. CSV documents are human-readable and can be processed by machines. CSV is used in log files and lists of events and have little overhead in size.

Raw logs contain security-related data and their structure vary in different systems. Logs can be easily read by humans but hardly parsed by machines because of structure diversity.

## 2.7 Data Exchange Models

Data exchange model determines how to exchange information between sharing peers. There are four data exchange models [100].

- **Source-subscriber**: Information is broadcasted from a sole server to a set of clients.
- **Peer-to-peer**: Information is produced and consumed by entities that are directly linked to each other.
- **Hub-and-spoke**: Information is transmitted to a central hub that manages the spreading to other members.
- **Push or Pull**: Information is pushed by a server to the clients or it is pulled by clients from a server. Push models are also called 'source/subscriber'.

Identifying the application domain and aims of information sharing is crucial while choosing the proper exchange model.

## 2.8 Threat Intelligence Tools

According to multivocal literature review conducted in [80] 22 threat intelligence sharing tools identified.

Accenture Cyber Intelligence Platform, Anomali ThreatStream, Anubis Networks Cyberfeed, Comilion, BrightPoint Security Sentinel, Eclectic IQ, Facebook ThreatExchange, Falcon Intelligence Crowdstrike, HP ThreatCentral, IBM X-Force Exchange, McAfee Threat Intelligence Exchange, Microsoft Interflow, ThreatCloud

IntelliStore, ThreatConnect, ThreatQ, ThreatTrack ThreatIQ are **commercial and closed source** applications.

AlienVault OTX and Soltra Edge are **closed source but free-to-use** tools.

MANTIS Cyber Threat Intelligence Management Framework, Malware Information Sharing Platform (MISP), Collective Intelligence Framework (CIF), Collaborative Research Into Threats (CRITs) are **open source** tools. As stated in [101], MANTIS is no longer be maintained and MISP is suggested.

This section compares and briefly describes remained three open source tools: Malware Information Sharing Platform (MISP), Collective Intelligence Framework (CIF), Collaborative Research Into Threats (CRITs).

### 2.8.1 Comparison of Threat Intelligence Tools

In this section we compared the aforementioned open source tools according to their support of other information sharing formats in Table 2.2.

Additionally, these tools are also compared with different aspects; software maturity, openness, readiness, usage and feeds. Table 2.3 summarize this comparison.

Table 2.2: Comparison of Cyber Threat Intelligence Tools 1

| Purpose | Standard Formats | CTI Tools | | |
|---|---|---|---|---|
| | | MISP | CIF | CRITs |
| Low-level Data | Pcap | – | – | + |
| | NetFlow, IPFIX | + | – | – |
| | CEF | + | – | – |
| Actionable Data | IDS rule | + | + | + |
| | YARA rule | + | – | + |
| | OpenIOC | + | – | + |
| | CybOX | + | – | + |
| | MAEC | – | – | – |
| | MMDEF | – | – | – |
| Vulnerability and Attack Data | CVE | + | – | + |
| | CWE | – | – | – |
| | CPE | – | – | – |
| | CVSS | – | – | – |
| | CWSS | – | – | – |
| | CAPEC | – | – | – |
| Threat Reporting Data | STIX | + | + | + |
| | IODEF | – | + | – |
| | MISP | + | – | – |

Table 2.3: Comparison of Cyber Threat Intelligence Tools 2

| Criteria/Tool | MISP | CIF | CRITs |
|---|---|---|---|
| Software Maturity Level | The project started in 2012 and development has actively went on with 13 project members and over a hundred contributors so far. 15 authors pushed 132 commits in one month period on June 2020. Source code is based on 72% PHP, 16% Javascript, 6% Python and forked by 861 times. | The project started in 2012 with version 1. After version 1 was finalized, Version 2(Massive Octo Spice) was developed at 2014. Version 3 (Bearded Avenger) has been actively in development since 2017 with one project member and 23 contributor. Only one commit is pushed in one month period on June 2020. Source code is based on 85% Python, 7% Shell, 4% HTML and forked by 54 times. | The project has been actively developed since 2014 with 45 contributors in total. Last commit is pushed at July 2019. Source code is based on 53% Javascript, 32% Python, 8% HTML, 6% CSS and forked by 239 times. |

*(continued on next page)*

Table 2.3 (continued)

| Criteria/Tool | MISP | CIF | CRITs |
|---|---|---|---|
| Openness | Open Source; allows commercial use, modification, distribution, private use with limited liability and warranty. (GNU General Public License) | Open Source; allows commercial use, modification, distribution, private use with limited liability, trademark use and warranty. (Mozilla Public License) | Open Source; allows commercial use, modification, distribution, private use with limited liability and warranty. (MIT License) |
| Readiness | Ready to use. | Ready to use. | Ready to use. |
| Usage | There is a detailed MISP book as user guide and installation can be done on Ubuntu, Debian, RHEL7, FreeBSD and CentOS distributions. | Installation is very easy and can be done only one Ubuntu distribution. There is a wiki page about the usage but poorly documented. | Installation is easy and can be done on Ubuntu and RHEL6 distributions. Documentation is present on wiki page and meet the needs. |
| Default Feeds | There are 63 intelligence feeds from different sources. | There are 28 intelligence feeds from different sources. | There is not any default feed for CRITs but another gathering tool can be used. |

In the following sections open source threat intelligence platforms are explained briefly.

### 2.8.2 Malware Information Sharing Platform (MISP)

MISP is an open source software for gathering, storing, correlating and exchanging threat intelligence in automated manner [90, 83]. It is developed by NATO Computer Incident Response Capability (NCIRC), Belgian Armed Forces and Luxembourg Computer Incident Response Center (CIRCL).

Indicators of Compromise data can be stored in a structured way so it can be exported to IDS, SIEM, STIX or OpenIOC format. Additionally different MISP instances can be synchronized.

As in stated in project website the primary goal is usability. Thus simplicity is the main principle while storing and sharing information about threats. There are also trusted sharing communities like FIRST, X-ISAC and NATO MISP communities which has different rules to join them.

MISP provides a RESTful API for data exchange purpose.

Feeds which contain indicators and automatically imported at regular intervals. They are remote or local resources and can be in three different formats; freetext, CSV and MISP standardized format. There are 63 default feeds [102] of a MISP instance.

#### *Support for standard data formats*

MISP support is explained in Section 2.4.5. For comparison see Table 2.2.

### 2.8.3 Collective Intelligence Framework (CIF)

CIF is an open-source cyber threat intelligence platform enabling you to fuse known threat data from many sources and use that data for identification, detection, and mitigation [103]. It was created by REN-ISAC (Research and Education Networking Information Sharing and Analysis Center). CIF mainly stores and processes IP addresses, domains, and URLs as threat data.

CIF aids you to parse, normalize, keep, process, generate, and distribute threat data. It supports ingesting many different sources of data sets such as feeds of malicious domains. Furthermore, threat intelligence data sets often have slight differences between them. It normalizes these data sets. It can be queried via a web browser, native client or using the API. CIF supports users, groups and API keys and each record can be tagged to be shared with specific group of users. It also supports creating new data sets from the stored threat intelligence.

CIF feed configuration files can be found in `/etc/cif/rules` directory [104]. Any other new feeds can be added with the correct yaml syntax used in these files. There are 28 default feeds of a CIF instance.

#### *Support for standard data formats*

CIF supports Snort and Bro rule formatted output. STIX is parsed in a limited extent by using a different module [105]. For comparison see Table 2.2.

### 2.8.4 Collaborative Research Into Threats (CRITs)

CRITs is a web-based threat intelligence tool that provides analysts to carry out collaborative research into malware and threats. Threat database is combined with an analytics engine to serve as both storage for attack data and a platform for analyzing malware [106]. CRITs offers a RESTful API for data exchange purpose. The project, which was first developed by MITRE, was made open source in 2013.

There is not any default feed for CRITs but another gathering tool Combine can be used to feed CRITs. Combine is an indicator collecting tool from publicly available sources [107].

#### ***Support for standard data formats***

CRITs accepts IDS rules (Bro and Snort) as signature types and has a module to analyze binaries with YARA rules. It also supports OpenIOC, CybOX and STIX version 1 with TAXII service feature. Additionally CRITs can use CVE data. For comparison see Table 2.2.

## 2.9 Blockchain Technology

Blockchain technology and related concepts are described in this section.

### 2.9.1 Blockchain

Blockchain is a distributed, immutable and transparent ledger(database) which holds transactional records and maintained within a decentralized network comprised of peer nodes. A replica of the ledger is kept on every node and a consensus protocol is used to verify transactions to be applied. After applied to the database, blocks — group of transactions — can not be altered since each one is hooked up to the preceding block via cryptographic hash functions. The consensus mechanism enables to eliminate the intermediaries or central authorities between participating nodes. Peer-to-peer communication and distributed copies of the data store provide high availability and eradicate single point of failure issues. Sharing all transactions and the world state — the final state — with all parties enables transparency. Immutability and integrity of shared data is achieved by using public key cryptography(digital signatures) to identify the legitimate peers and linking successive blocks with hashing. Moreover, activities in the network can be automated by adopting smart contracts.

### 2.9.2 Permissionless and Permissioned Blockchains

As a first implementation, Bitcoin allows mutually mistrusting parties to carry out financial transactions without any trusted third party intervention [108]. It also offers data store integrity and transparency properties inherent from blockchain technology itself. These features led to the use of technology for other purposes and permissioned blockchains emerged.

Unlike Bitcoin's permissionless blockchain, where anyone can join pseudo-anonymously, only authorized participants can join to the permissoned network [109]. Permissionless mode is fully decentralized but permissioned one requires a central entity which gives permission to nodes to participate, therefore it is partially decentralized. While permissionless blockchain include economic incentives to join and comprised of mutually mistrusting nodes, permissioned blockchain entities have a common goal as an incentive and they do not fully trust each other. For enterprise use case, permissioned network is appropriate, as participants must be identifiable, and business transactions and data must be private and confidential.

### 2.9.3 Consensus Mechanism

As a predefined rule set, the consensus mechanism is used to validate all transactions and reach an agreement between all nodes. Consensus protocol is vital to keep the ledger in a single state, to manage fault tolerance and malicious behavior. The mechanism should be selected appropriately in accordance with the mode of operation of the network. For instance, permissionless networks with pseudo-anonymous nodes need to use proof-of-work (PoW) which ignores trust and demands CPU power. However, mechanisms like crash fault tolerance (CFT) and byzantine fault tolerance (BFT) provide high transaction performance and therefore suit the enterprise networks where the participants are known. A CFT based protocol is resilient to the crash of (n-1)/2 nodes but it does not tolerate any subverted(malicious) nodes [110]. On the other hand, a BFT based consensus is both resilient to crash and subversion (n-1)/3 nodes but it is slower [111].

### 2.9.4 Decision Making

Generally, using a blockchain solution is appropriate when multiple parties reciprocally do not trust each other, do not want to use an online trusted third party, and need to modify and share the state of a system [109]. In our scenario, CTI sharing parties have a common intention; sharing to stop the adversary, but they do not completely trust each other. While deciding the blockchain solution, we follow a methodology introduced in [109] to identify whether a blockchain is practical for CTI sharing problem requirements.

To ease the decision making, we answered the questions in the flow chart shown in Figure 2.3. This flow chart is utilized while deciding if a blockchain is a practical solution to CTI sharing. First of all, a data store is a must for exchanging threat information. Similarly, considering database writers, multiple parties interact with and write to the shared database. Trusted third party acts as a certificate authority for identifying participants and it isn't always online. Members are known but they do not fully trusted. Lastly, public verifiability is not desired, because readers should also be restricted. As a result private permissoned blockchain is a suitable solution for our use case.

Open source solutions have been examined as candidates for our use case. Hyperledger Fabric [112], R3 Corda[113], and Quorum [114] are designed as private blockchain solutions. We can add Ethereum [115] to this list since it can also be configured to run in private mode. Therefore, we have addressed these four solutions during the selection phase.

While Fabric and Ethereum are designed flexibly to fit every industry, R3 Corda and Quorum are originally designed for the financial sector [116, 117]. Another characteristic is programming language support for smart contracts, which affects the ease of development. Distributed smart contracts can be written in general-purpose programming languages in Hyperledger Fabric, Quorum, and R3 Corda. However, Ethereum only allows us to write the smart contracts in Solidity, a domain-specific language. Regarding the consensus mechanisms, private blockchain solutions do not

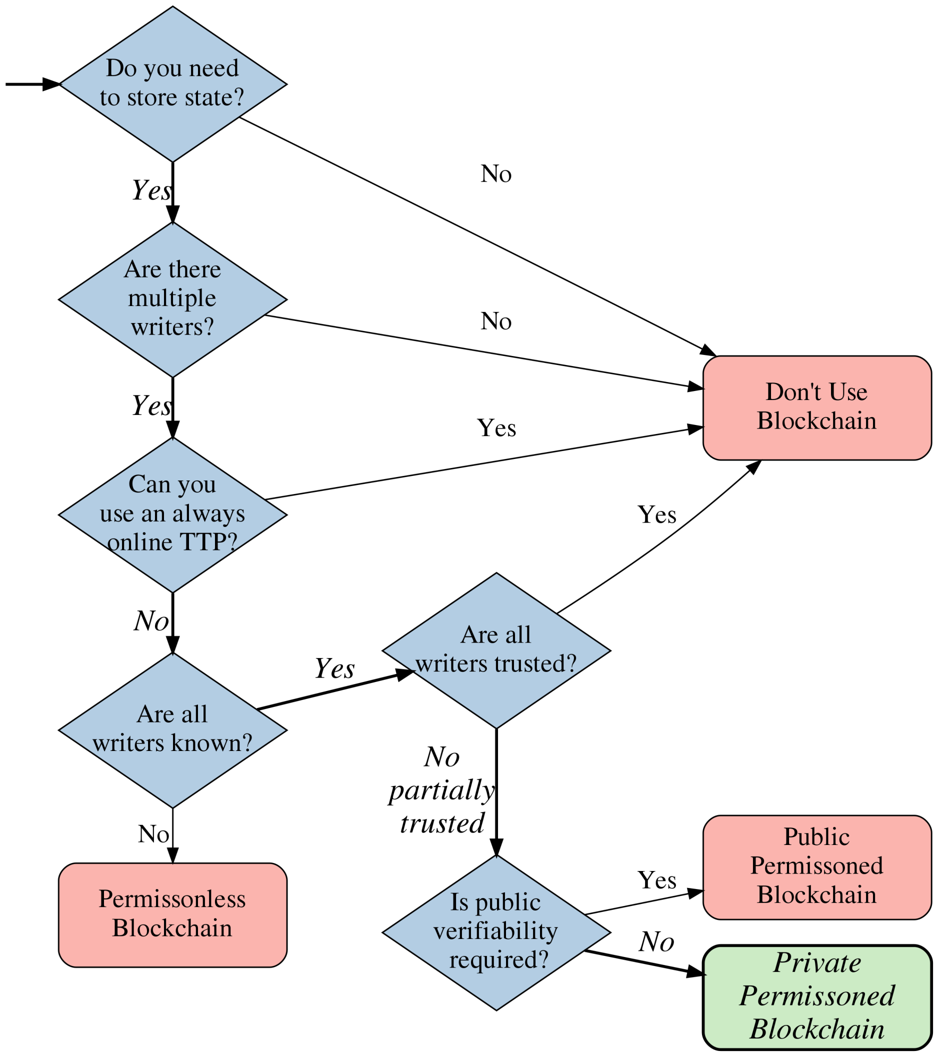

Do you need to store state?
Yes
No
Are there multiple writers?
Yes
No
Can you use an always online TTP?
No
Yes
Don't Use Blockchain
Are all writers known?
Yes
No
Permissonless Blockchain
Are all writers trusted?
Yes
No partially trusted
Is public verifiability required?
Yes
No
Public Permissoned Blockchain
Private Permissoned Blockchain

Figure 2.3: Decision Making Flow Diagram
Adapted from [109]

need costly PoW protocols. Therefore, each solution provides options with different resiliency properties. Hyperledger Fabric employs CFT-based Raft consensus protocol, and it also allows to use of BFT-based solutions. Quorum has Raft, Istanbul Byzantine Fault Tolerance (IBFT), and Clique Proof of Authority (PoA) consensus protocol options. R3 Corda also allows BFT and CFT based consensus protocols. On the other hand, private Ethereum blockchain can operate on Clique PoA protocol. Lastly, none of the other blockchain options need native cryptocurrency to run smart contracts, but Ethereum needs Ether for charging computational costs.

We have not chosen Quorum and Corda as they focus on financial applications and have less community and research support than Fabric. On the other hand, Ethereum has lower throughput and higher latency than Hyperledger Fabric [118, 117]. Additionally, there are other important factors for choosing HLF; any currency is not needed and general-purpose languages are supported. Last but not least, various research is ongoing about improving the Hyperledger Fabric security, performance and scalability issues [119, 120, 121]. Properties of discussed private blockchain solutions are compared based on different aspects in [117], and the result of the research analysis is presented in Figure 2.4.

In summary, we have selected Hyperledger Fabric because of above reasons;

- Modularity that fits for almost all use cases,
- Strong privacy and confidentiality properties,
- High throughput and low latency,
- Strong development and research community,
- Usage of general purpose languages in smart contract development,
- No need for any currency.

## 2.10 Hyperledger Fabric

Hyperledger Fabric framework and components are presented in this section.

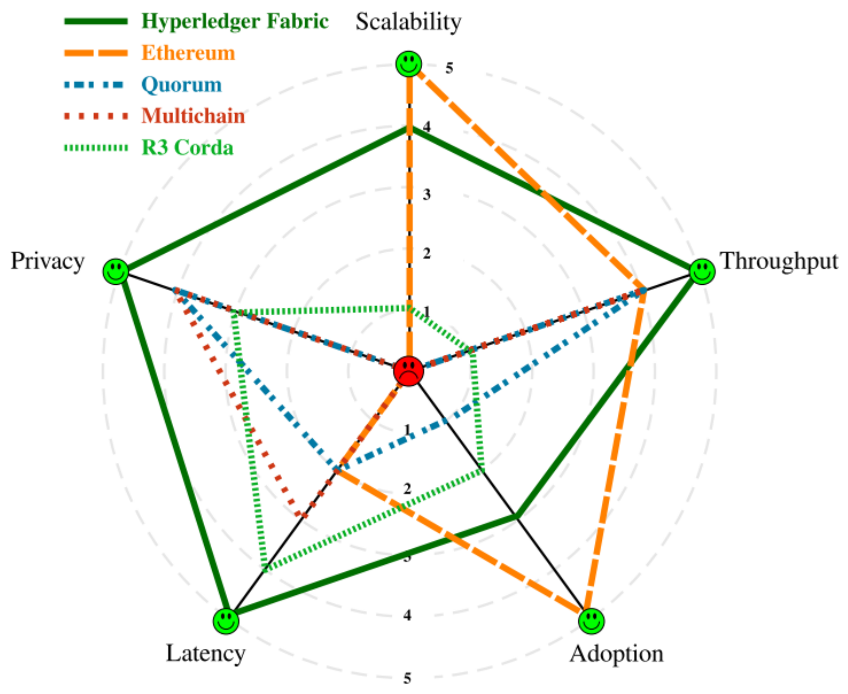


Figure 2.4: Comparison of Private Blockchain Solutions
Retrieved from [117]

### 2.10.1 Hyperledger Fabric Platform

We implemented our model by using Hyperledger Fabric, a modular and extensible permissoned distributed ledger technology (DLT) platform, designed for enterprise use from the beginning [112, 122]. It is one of the open source projects under the Hyperledger umbrella project governed by Linux Foundation. There are nearly two hundred engineers from over 35 organizations involved in the Hyperledger Fabric development. The version 1.0 Fabric was released on July 11th 2017. It is production ready at this point and evolving at a very rapid pace. As of December 2019, stable version 1.4 and beta version 2.0 is released.

In general, Hyperledger Fabric is a framework that allows the businesses to create decentralized applications for exchange of value. It has several advantages over permissionless blockchain networks;

- It provides pluggable infrastructure and application building blocks that enable more efficient platform customization for different circumstances. For example, one can choose the most effective consensus protocol suitable to the use case.
- It supports the use of general-purpose programming languages — Java, Go and Node.js — to develop smart contracts.
- There is no need for a cryptocurrency that leads to costly mining process.
- It also provides confidentiality and privacy of chaincodes and transactions.
- Unlike permissionless networks, resource intensive validation schemes aren't used which leads to better transaction confirmation latency.

Details about architecture and components are described in the following subsections.

### 2.10.2 Assets

Assets represent some kind of value that can be exchanged on the blockchain systems. Any object which has a value in the real world may be represented as an asset on Fabric as long as it can be represented digitally. There are tangible and intangible assets. While hardware is an example of the former, threat information is an example of the latter. Asset states are stored in the ledger as key-value pairs in JSON or binary format. Through well-defined transactions coded in chaincode, Hyperledger Fabric enables us to change the states of these assets. State-changing transactions are also recorded in the ledger immutably.

### 2.10.3 Chaincode

Chaincode, also known as Smart Contract, defines the structure of the asset(s) and the transactions that can be executed for altering the asset(s). It has all of the business logic needed for the transactions. To query and change(commit) key-value pairs, pre-defined, agreed-upon rules are applied by chaincode.

In order to announce a transaction as valid, there is an endorsement policy that specifies which entities in a blockchain channel must approve a transaction generated by a particular chaincode. The specification of the endorsers uses logical expressions, such as a set of peers; $(Org1Peer \wedge Org2Peer)$. This policy means two endorsement is needed from peers belonging to Org1 and Org2.

### 2.10.4 Ledger

Ledger is a data structure that keeps track of all transactions. It also records the state changes taking place in the assets as a result of execution of these transactions. There is one logical ledger per channel and it is distributed; peers, which are channel members, have a replica and maintain a copy of the ledger.

The ledger is comprised of two distinct parts; a world state and a blockchain.

- **World state** stores the current values of the asset states. LevelDB and CouchDB are two current options of databases on which world state can be stored. While the former is the default option and embedded within the peer node, the latter runs externally and supports rich JSON-based queries.

- **Blockchain** stores the immutable, sequenced transaction records which determines the current world state. There is almost little need for the query operations, and the main actions are additions to the end of the chain. Due to these characteristics, the blockchain designed to be always stored in a file, unlike the world state.

### 2.10.5 Identity

In a permissioned network, anonymous access to blockchain applications is not allowed, thereby identity assignment to participants is necessary. Blockchain application defines the roles that are assigned to the participants and access is granted or restricted by way of these roles through validating the identities. When a participant identity is created, the certificate is issued to the participant. Anytime a transaction is initiated by a participant, its private key used for signing the transaction. Any component in the network can validate the authenticity of the transaction by using the participant's public key. It is not only the participants but also the infrastructure components are assigned an identity by way of certificates. It prevents a scenario where hackers can add a server to disrupt the network or to make an attempt to manipulate the transactions.

### 2.10.6 Membership Service Provider

The identities are created and managed by using Membership Service Provider (MSP). By default, the Public Key Infrastructure (PKI) model is adopted and X.509 certificates are utilized for identification. Trusted Root CAs and Intermediate CAs are determined by MSP while defining a trustworthy network. Certificates follow the typical process of issuance and the revocation by the certification authorities in the network.

### 2.10.7 Certificate Authorities

The certificate authority (CA) is an implementation of the MSP. In addition to allowing the use of external and commercial CAs, Hyperledger Fabric provides the Fabric CA. Required key pairs and certificates to manage identities can be created using this solution during the development phase.

### 2.10.8 Organizations

Organizations, also known as members, are legally separate or independent entities that have decided to join blockchain network. Each organization has its own MSP which has to be added to the blockchain network before the network participation. Members can manage the identities within their organization via their MSP. They can create a participant and an infrastructure component identity within their organization. This aspect remove the dependency on a single centralized certification authority.

### 2.10.9 Nodes

Nodes are a communication entities in a blockchain network. They connect to other nodes to form a blockchain network, and use a peer to peer gossip protocol for keeping the distributed ledger in sync across the network.

Since the nodes are the communication endpoints, the applications used by the participants connect to the network through nodes. The identities of nodes and participants are different. Node's certificate is used by the network to check if the node is trusted or not. However, participant's certificate is used for signing the transactions.

In Hyperledger Fabric, unlike the permissionless blockchain technologies, all nodes are not equal. There are three distinct type of nodes;

- **Client nodes** that applications use for initiating the transactions through transaction proposals.

- **Peer nodes** that keep the ledger in sync across the network, execute transaction proposals and validate transactions. However, only a subset of them, endorsing peers, execute transaction proposals. Endorsing peers are determined by the chaincode endorsement policy.

- **Orderers** that are in charge of arranging all transactions. They put in order transactions to create transaction blocks and distribute fresh blocks to the channel peers. They are used in the Fabric consensus mechanism and aren't involved in the execution and verification of transactions.

A basic Fabric network consisting of two organizations and associated CAs is shown in the Figure 2.5. The client node is permitted to communicate with the channel based on its certificate issued from a member organization of the network.

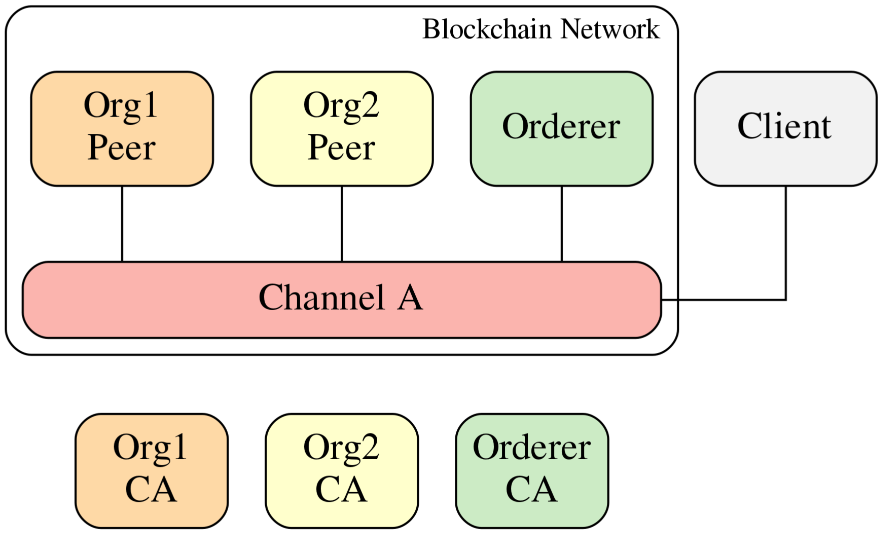


Figure 2.5: Basic Hyperledger Fabric Network

### 2.10.10 Consortium

Organizations other than the Orderers constitute the consortium that must be created per-channel basis. While multi-consortia blockchain networks are an option, there are often networks with one consortium. All organizations to be added to the channel must be a part of that channel's consortium during the channel formation. Nevertheless, an organization not defined in a channel consortium is also allowed to be added to a preestablished channel.

### 2.10.11 Channel

Channel design of the Hyperledger Fabric provides confidentiality. An isolated channel can be formed between a subgroup of members who have permission to query and commit specific transactions. It maintains the confidentiality and privacy of the chaincode and the ledger by giving authorization only to the authentic channel participants. A peer connected to one channel cannot access to the ledger and the chaincode of another channel of which peer is not a participant.

Consider the example in Figure 2.6 where there are three members and they have decided to launch a blockchain network. Here as you can see there is Channel A and a ledger and the chaincode that is available on that channel to all three members. Now let's say Org1 and Org3 decided to have some kind of deal where they want their transactions to be private. Therefore, they can create a private channel. In this scenario Org1, Org2 and Org3 are connected to the common channel (Channel A) whereas Org1 and Org3 are also part of that common channel. Along with that they have created a private channel (Channel B). The ledger and chaincode for that private channel is independent and isolated from the ledger and chaincode for the common channel.

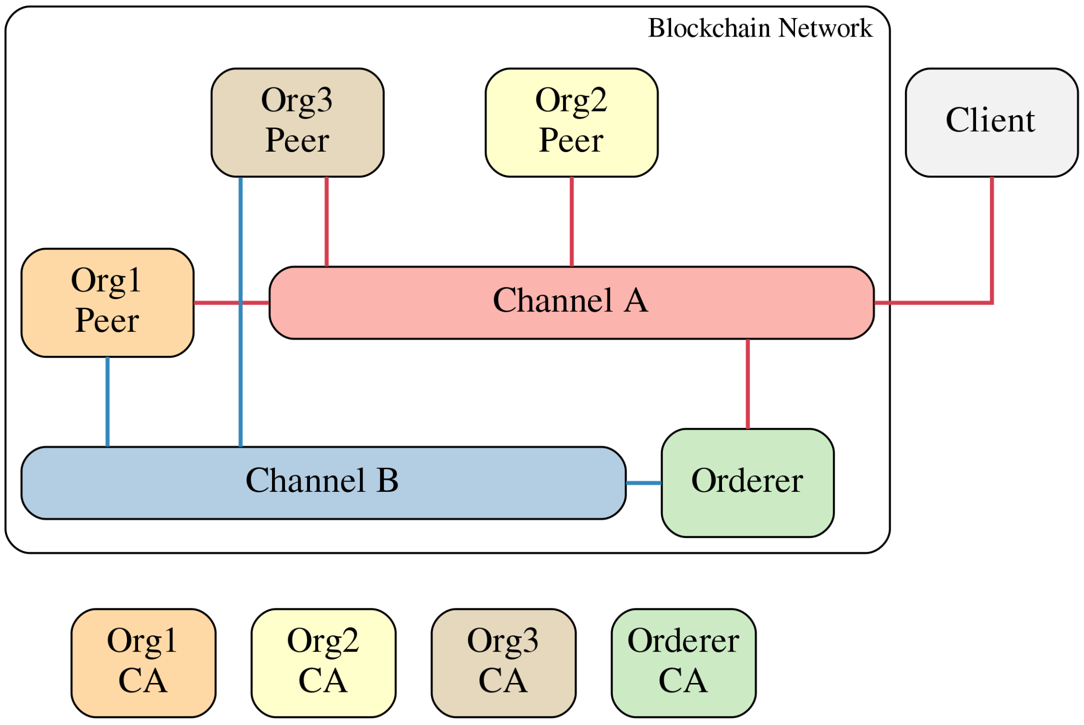


Figure 2.6: Hyperledger Fabric Network With Two Channels

Chaincodes and ledgers distributed on peers are not shown in this figure for the sake of simplicity.

## 2.11 Research

Even though there are a lot of products that consume and produce threat intelligence, research side is very deficient. We conducted a literature review for understanding the complete picture comprehensively in this area.

Kampanakis [123] and Dandurand et al. [48] give explanatory view of the sharing formats. Most of these formats are competing with each other and some of them are not relevant nowadays.

Additionally some articles handle the issue comparatively. Steinberger et al. [124] classifies information sharing formats and communication protocols, and compares sharing formats in context of interoperability with flow based data and machine read-ability. Mavroeidis and Bromander [125] evaluate existing standards with their proposed threat intelligence model and emphasize the lack of threat ontology. Menges and Günther [126] introduce an incident reporting process model to conduct a comparative analysis of incident reporting formats.

Zibak and Simpson performed literature review to categorize the benefits and the challenges of information sharing, and surveyed cyber security professionals from different sectors [8].

Dandurand and Serrano defined the high-level requirements of threat intelligence platforms as promoting information exchange, allowing automation, and facilitating the production, refinement and analysis of data through collaboration [127].

Sillaber and Sauerwin examines the factors that affect shared data quality and propose recommendations according to their findings [128].

Appala et al. [129], developed XMPP based actionable threat intelligence platform that provides security and extensibility.

Several researchers perform indepth literature survey and compare threat sharing tools [84, 100]. Moreover, Sauerwin et al. [80] analyzes and classifies information sharing tools.

De Fuentes et al. [130], propose PRACIS, a privacy-preserving protocol for CTI sharing. Authors offer a new security layer when the network is unreliable and provided by a third party.

There are a few researches that propose new CTI sharing model based on different types of blockchain technology. Riesco et al. [131], proposed Ethereum based public permissionless blockchain network model which can be incentivized with Ether or CTI token. They have also conducted simulations to demonstrate its benefits. Homan et al. [132], proposed Hyperledger Fabric based permissioned blockchain network model but performance characteristics are not discussed. Hajizadeh et al. [133], presented a Collaborative DDoS attack mitigation system based on Hyperledger Fabric, and conducted performance experiments.

This research aims to clarify the CTI landscape, compare most relevant reporting standards. As far as we know, there is not an existing research that compares threat reporting formats according to their support of other data sharing formats. Additionally, this study propose a new CTI sharing network model based on permissioned blockchain technology — Hyperledger Fabric. Finally, performance aspects of proposed model is discussed.

## 2.12 Challenges

Even though threat sharing has undeniable benefits while combating cyber attacks, it has also many drawbacks. These shortcomings are often linked with each other, but they are categorized for research purposes. In the sequel, they are briefly explained.

### 2.12.1 Lack of Interest

According to Ponemon Institute's survey conducted with 1200 IT security specialists in the US and EMEA, number one reason for not exchanging threat intelligence is perception that there is no benefit to the organization [134]. Respondents also partially participate because of this reason. Furthermore, sometimes organizations think that information about an attack attempt is not worth to share since it is not successful [19]. However, an unsuccessful attack to one company might be successful if directed towards another in the same sector. ENISA conducted a survey among 22 CERTs across Europe, and lack of interest is one of the key reasons for information sharing [135]. Some organizations still have a culture of blame - if something happens then it's clearly someone's mistake. It causes a natural instinct not to share [136].

### 2.12.2 Lack of Automation

Another hindering reason is using manual methods instead of automated tools. NIST suggest to use common data formats and protocols to enable automation. However adopting certain standards can bring certain time and resource load [137]. Approximately half of security practitioners use manual methods and more than half of them are not satisfied because information is not timely [134]. As reported by ENISA, technical barriers — automated sharing and data quality — are the most common obstacle to information sharing [135]. Sometimes one part of a dispersed corporate has experienced an unsuccessful attack months ago but hasn't informed the other part and same attack would be successful in that part of the company. Because it has no threat sharing tool to communicate even internally [136]. It is obvious that automated tools with common language can improve the interoperability and timeliness, as a result the quality of actionable threat information.

### 2.12.3 Legal Uncertainty

Legal issues are another intimidating aspect of information sharing [135, 100]. CTI exchange is done both nationwide and globally so various laws and regulations have to be considered. Organizations are not sure about which information can be exchanged to avoid juridical questions regarding privacy protection. CTI can contain personally identifiable information and sensitive data should be sanitized before exchange [123]. For example; IP addresses are considered personal information in some circumstances [138]. In some countries, sanctions are put in place for organizations that do not share security breaches with the authoritative bodies. Regulative rules may be different for sharing parties not only in different countries but also in the same country among different organizations. The types of information an organization shares, can be restricted by the executive and legal units of that organization [137]. Appropriate restrictions are necessary but unreasonable ones are reduce the quality of shared data.

### 2.12.4 Lack of Trust

Another important concern is trust within sharing community. It is essential since stakeholders have a need to rely on each other and the infrastructure [137, 130]. Without it nobody will participate and share pertinent information. However, lack of trust among stakeholders is often ignored [80, 136]. The fact that sharing platforms are mostly closed source software also undermines trust [80]. Furthermore, disclosing the weaknesses of the corporate while exchanging threat intelligence prevents to participate [19]. There are several solutions to this problem; ENISA recommended not to disclose competitive data while sharing of Modus Operandi [139]. There is also a possibility that competitor organizations may share misleading threat information. According to Ponemon survey [134], companies prefer not to share threat data directly but via a trusted third party.

### 2.12.5 Safeguarding Sensitive Data and Privacy

While insecure sharing infrastructure provided by a third party is being used, privacy is vital to support collaboration [130]. Furthermore, disclosure of sensitive data may result in violations of the law, financial loss and loss of reputation [137]. Threat actors can gain information about cyber security capabilities of sharing organization via shared security event data and change their TTPs. Anonymization of shared data is suggested to limit attribution. However, limited attribution may reduce the quality of threat information [137].

### 2.12.6 Low Quality of Shared Data

The problem of data quality is important because cyber security cannot be based on old, unreliable information. An organization must verify that the information is accurate, relevant, and the risks about the threat are well understood before acting on that [137]. Even there are some sharing efforts between industry, a lot of what is shared is outdated [136]. Users are dissatisfied about low quality data and do not trust the commercial data providers because of inefficiency and out of date IOCs [134, 136]. Quality problems worsen when the number of stakeholders and integrated data sources increases [128]. Relevance aspect should also be considered. It is harder to find right intelligence in time as your platform is included data relevant to other sectors. In order to handle quality concerns [128, 100];

- Shared data should be relevant, timely, represented in a structured way, enriched with additional metadata,
- Platform should have convenient automation, distillation, correlation mechanisms,
- Data entry rules and in which case the data will be shared should be determined between peers in advance.

### 2.12.7 Inevitable Costs

Engaging costly resources — monetary and personnel — for information exchange is usually a concern. Expensive commercial solutions and hard to manage open source solutions are two options which have different type of costs [100, 136]. Ponemon's survey stated that 21% of respondents consider costs are one of the reason for not participate a exchange program [134].

As a result, institutions that prefer to not to participate in exchange programs are thwarting the information sharing efforts on which cyber security depends. However, proposed model is trying to mitigate the shortcomings and foster CTI sharing. Table 2.4 summarizes the mentioned challenges.

## 2.13 Proposed Solution

The purpose of this study is to present reasonable solutions for some unresolved challenges in CTI sharing as shown in Table 2.4.

A CTI sharing model based on permissioned blockchain is proposed, and several performance and functionality experiments are held to determine the advantages and constraints of the proposed model. As a result, the features of the proposed model that can solve the sharing problems are presented below, together with the justifications.

Table 2.4: Challenges of CTI Sharing

| ID | Description | References |
|---|---|---|
| 1 | Lack of interest and sharing is not believed to be useful | [19, 134, 135, 136] |
| 2 | Lack of automation | [134, 135, 136, 137] |
| 3 | Legal uncertainty and potential liabilities | [100, 123, 134, 135, 137] |
| 4 | Lack of trust between peers and infrastructure | [19, 80, 130, 134, 136, 137, 139] |
| 5 | Safeguarding sensitive data and privacy | [130, 137] |
| 6 | Low quality (reliability, accuracy, relevance and timeliness) of shared data | [128, 134, 136, 137] |
| 7 | Inevitable costs | [100, 134, 136] |

### 2.13.1 Security and Trust

We have chosen Hyperledger Fabric, an open-source permissioned private blockchain solution that only authorized users can join and is better suited for business-to-business (B2B) applications. Hyperledger Fabric uses PKI for participation, supports Elliptic Curve Digital Signature Algorithm (ECDSA), communication can be secured with TLS, and ledger data can be stored with encryption. Moreover, the chain code cannot be modified without the consent of the channel members. Thus, it ensures the reliability of previously agreed business logic. These aspects of the HLF network can solve problems of trust and sensitive data protection. Stakeholders can rely on this infrastructure while sharing threat data.

### 2.13.2 Availability

Hyperledger Fabric has distributed ledgers in each peer node across the network, and eliminates the single point of failure problem. This feature improves availability, eliminates data loss, and therefore increases the trust of the infrastructure.

### 2.13.3 Accountability

The non-repudiation of shared data among participants is vital while two competent organizations are in the network. Thanks to the immutability and transparency features of the blockchain, participants can see who actually produces/consumes what and when. This feature provides accountable sharing and there could be no chance of denying or falsifying any shared threat data. Therefore, the trust of the participants in each other and in the infrastructure increases.

### 2.13.4 Efficiency

Automation can be enabled by using standardized data model, MISP, and smart contracts. MISP automation API can be used to generate signatures for different security devices like IDS/IPS, APT scanners, etc. Furthermore, course of actions can also be implemented within the smart contracts. Automation allows participants to get the threat data on time. These features address the lack of automation and data quality issues. We have tested this feature by automating the check of the TLP labels of the events to ensure secure sharing.

### 2.13.5 Low cost

Decentralized sharing network is backed by different organizations with burden-sharing principle. The cost of each node is sustained by the organization that owns the node. This aspect of the model decreases the cost issues.

### 2.13.6 Privacy

Within the same Fabric network, multiple channels can be utilized while sharing privacy-sensitive threat data. Compartmentalization of the network allows us to share based on the need-to-know principle. We have tested this aspect of the network in our experiments and observed its functionality.

Properties of our proposal resolve some of the challenges mentioned in Section 2.12. These solutions are mapped to the challenges in the Table 2.5 according to the benefits they provide.

Table 2.5: Solutions of Proposed Model

| ID | Provided Solution | ID of Challenge in Table 2.4 |
|---|---|---|
| 1 | Trusted and secure network provided by blockchain | 4,5 |
| 2 | Availability of distributed ledgers | 4,5 |
| 3 | Accountable transactions | 4 |
| 4 | Efficient (automated) data sharing | 2,6 |
| 5 | Low cost with burden-sharing | 7 |
| 6 | Privacy aspect of Hyperledger Fabric | 5 |

# CHAPTER 3

# METHODOLOGY

This chapter shows the followed methodology of this study, the configurations and choices that we have made. We also discuss how to test our network model.

## 3.1 System Goals and Requirements

We proposed a blockchain network to share CTI data between organizations. Blockchain network should provide an auditable and secure sharing environment and automate the sharing process among organizations.

The following requirements will be our system goals due to the importance of the CTI data.

### 3.1.1 Security and Trust

Sensitive information can be leaked through CTI data. Sharing critical data using a public blockchain system is not an appropriate solution. As a result a permissioned blockchain, Hyperledger Fabric, is used and access control is enforced.

### 3.1.2 Integrity and Accountability

Data integrity must be maintained in order to accurately respond to attacks. After the data is shared, it should not be possible to be changed by unauthorized parties.

Blockchains inherently provides data integrity through immutable hashed blocks and decentralized storage.

### 3.1.3 Availability

Organizations may need data immediately to respond to attacks. For this reason, the availability of data and network is very important. Distributed ledger technologies like blockchains are appropriate solution for availability.

### 3.1.4 Automation

Cyber attacks often require urgent action and automation. Automation of data requiring urgent processing depends on its standardization. Furthermore, structured data can also be easily consumed by machines. In our proposal we used JSON based MISP data standard while storing the CTI data in distributed ledgers. Another feature of our solution is chaincode which automates the implementation of the essential business processes.

### 3.1.5 Privacy

Data privacy is needed while sharing sensitive information. Organizations demand to share different threat data with different partners to comply with the need-to-know principle. Our solution provides channel-based division of the network, and different parties can share different types of threat data while using the same infrastructure.

### 3.1.6 Client Application

A client application is needed to simulate different actors in the network. Therefore, we have developed a minimal Node.js web application enabling users to connect with the sharing network through HTTP requests. RESTful APIs are provided for different chaincode functionalities via Fabric SDK which allows the client applications to submit transactions to the blockchain network and execute the chaincode. A separate client application is assumed for each member accessing the sharing network.

## 3.2 Implementation

This section presents the implementation process of our blockchain network and our sharing application in detail. The steps for developing and running a Hyperledger Fabric network is going to be described. Due to the fact that Hyperledger Fabric is still in early stages, our implementation is very time-consuming. Vague error messages make it difficult to detect problems. We also used some customized scripts in order to establish networks with different configurations easily.

### 3.2.1 Environment Setup

Hyperledger Fabric blockchain network development require some prerequisites and binaries already installed [140]. Table 3.1 shows the environmental setup. We have installed Docker, Docker Compose and Fabric binaries on Ubuntu base system which runs on a laptop with eight Intel® Core™ i5 - 8250 CPU @ 1.6GHz processors and 20 GB RAM. Docker and Docker Compose is used to generate and operate the entities in the network. We have used Node.js for chaincode development. Due to the fact that Fabric uses Go for its components, this language is also required.

Table 3.1: Environmental Setup for Hyperledger Fabric Network

| System&Tool | Version |
|---|---:|
| Operating System | Ubuntu 18.04 |
| Hyperledger Fabric | 1.4.4 |
| Docker | 19.03.6 |
| Docker Compose | 1.25.4 |
| Node.js | 12.16.1 |
| Go | 1.14.4 |

Fabric binaries are downloaded with the command in Listing 3.1. First version number specifies the version of Fabric binaries and Fabric docker images, second one defines the Fabric CA docker image version and the last one specifies the CouchDB docker image version.

**Listing 3.1** Download fabric binaries

```
curl -sSL https://bit.ly/2ysbOFE | bash -s -- 1.4.4 1.4.4 0.4.18
```

In order to setup the blockchain network and configure the entities, yaml files are used extensively by both docker-compose and fabric binaries. To produce crypto materials, the corresponding yaml file is passed as a parameter to `cryptogen` as shown in Listing 3.2.

**Listing 3.2** Generate crypto materials

```
cryptogen generate --config=./crypto-config.yaml
```

To create orderer block and channel configuration transaction file, `configtxgen` binary is used as shown in Listing 3.3. This binary is implicitly using the `configtx.yaml` file. Profile parameters of these commands are same as the orderer and channel profile settings in the `configtx.yaml`.

**Listing 3.3** Create the orderer block and the channel configuration transaction

```
# create the orderer genesis block
ORDERER_PROFILE="OrdererGenesis"
SYS_CHANNEL="cti-sys-channel"
configtxgen -profile $ORDERER_PROFILE -channelID $SYS_CHANNEL \
            -outputBlock ./channel-artifacts/genesis.block

# create the channel configuration transaction
CHANNEL_PROFILE="ChannelCTI"
CHANNEL_NAME="channelcti"
configtxgen -profile $CHANNEL_PROFILE \
            -outputCreateChannelTx ./channel-artifacts/$CHANNEL_NAME \
            -channelID $CHANNEL_NAME
```

After creating orderer block and the `channel.tx` file, `docker-compose` command in Listing 3.4 starts the network entities; orderer, peer, fabric-ca, cli and couchdb instances. The settings of these components are configured in the `docker-compose.yaml` file.

**Listing 3.4** Start the blockchain network components

```
docker-compose -f docker-compose.yaml up -d
```

When all of our network components are running, we access the CLI container with `docker exec -it cli bash` command and act on behalf of peers by changing environment variables. First, we create the channel and make all peers to join to this channel as demonstrated for one peer in Listing 3.5.

**Listing 3.5** Create and join to the channel

```
# create the channel
ORDERER_ADDRESS="orderer.cti.com.tr:7050"
CHANNEL_NAME="channelcti"
peer channel create -o $ORDERER_ADDRESS -c $CHANNEL_NAME \
                    -f ./channel-artifacts/$CHANNEL_NAME.tx
# join to the channel
peer channel join -b $CHANNEL_NAME.block
```

Lastly, the chaincode is needed to be installed on all of the peers, and instantiated on the channel once. Commands used in these steps are presented in Listing 3.6. These commands are executed from cli docker container. The chaincode instantiation generates a docker container and code runs in this isolated environment. While instantiating the chaincode, endorsement policy is determined by using a simple logic expression as a parameter. The example shows an endorsement policy where at least one member from both organizations have to sign for the transaction to be valid. Once the chaincode is instantiated successfully, the peers can begin to transact and we can start our HTTP server for interacting with the network.

### 3.2.2 System Architecture

In this section we present the components in our base architecture prior to testing the network. At the evaluation phase we have changed the architecture and different configuration parameters to see the effects on the network. The overview of system architecture is depicted in Figure 3.1.

**Listing 3.6** Install the chaincode on one peer and instantiate on the channel

```
# install the chaincode
CHAINCODE_NAME="CyberEvent"
VERSION="1.0"
LANGUAGE="node"
CHAINCODE_SRC_PATH="/opt/gopath/src/github.com/chaincode/"
peer chaincode install -n $CHAINCODE_NAME -v $VERSION \
                       -l $LANGUAGE -p $CC_SRC_PATH
# instantiate the chaincode
ORDERER_ADDRESS="orderer.cti.com.tr:7050"
CHANNEL_NAME="channelcti"
peer chaincode instantiate -o $ORDERER_ADDRESS -C $CHANNEL_NAME \
                           -n $CHAINCODE_NAME -v $VERSION -l $LANGUAGE \
                           -c '{"Args":["instantiate"]}' \
                           -P "AND ('Org1MSP.peer','Org2MSP.peer')"
```

**Hyperledger Fabric Network** stores chained transaction records on distributed ledgers and access control is applied based on user identity. The network consists of following components.

- **Orderer:** This component is responsible for ordering transactions and provides a consensus mechanism which keeps the ledger in sync. When it receives update transaction proposals, it put in order and packs them in blocks. Then valid blocks are distributed to anchor peers on the channel. Although we used only one Orderer with Solo consensus mechanism, this is not recommended in a production environment.

- **Peers:** Two peers, one of them anchor, are deployed for each organization in the network. Anchor peers are discoverable by Orderer and peers in different organizations through gossip data dissemination protocol. They receive updates and broadcast them to the other peers in their organization. In our setup anchor peers are also endorsing peers which execute transaction proposals.

- **CAs:** They carry out the task of distributing the certificates to network participants. Then these certificates are used to authenticate members. A Fabric CA server instance is being run by each organization in our network. Each CA server issues certificates with previously generated cryptographic materials. Fabric has also the ability to interoperate with real certificate authorities in

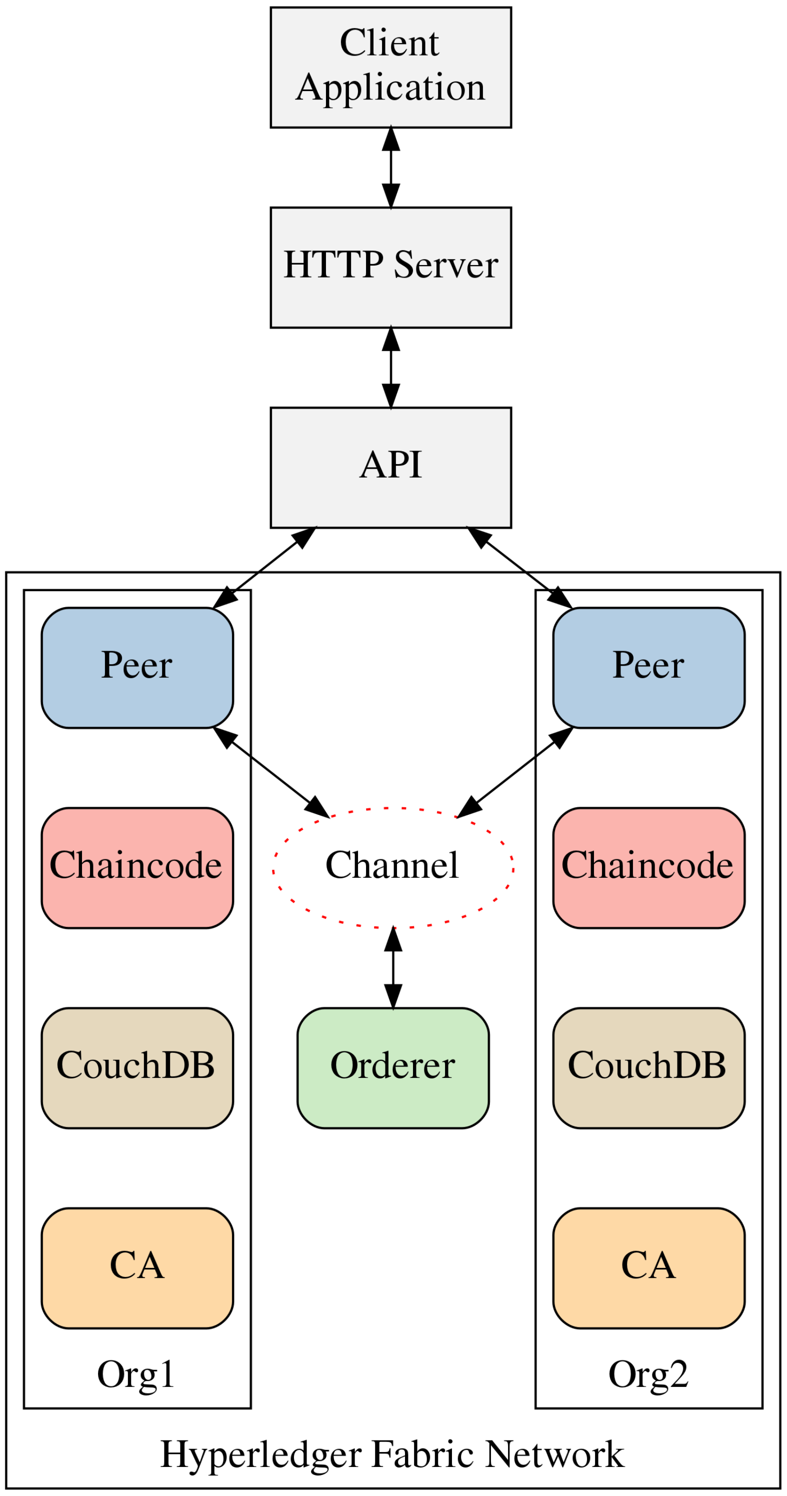
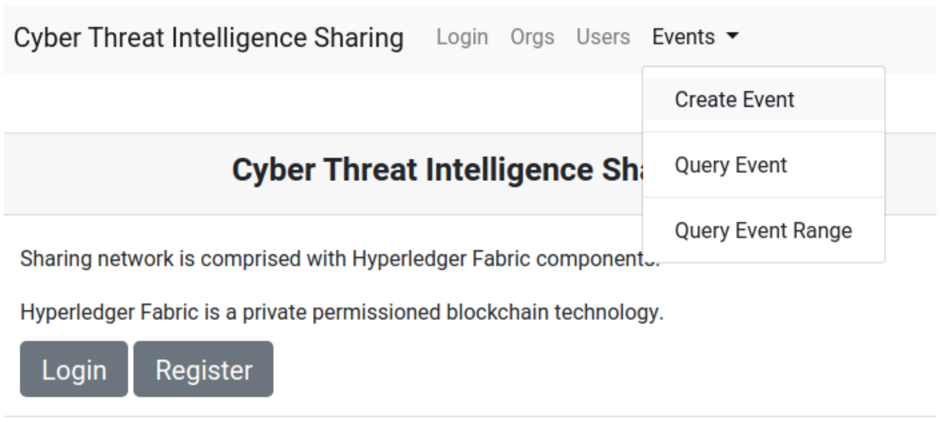


Figure 3.1: System Architecture

real-world deployments.

- **CouchDBs:** In order to store ledger state there are two options, LevelDB and CouchDB. In our network we used CouchDB as state database and a CouchDB instance is running for every peer.
- **Chaincode:** After the instantiation of installed chaincodes these components are activated and chaincode runs in this isolated environment.
- **API:** Available Fabric SDKs allow client applications to connect with the blockchain network. SDK developed for Node.js is used in our solution.

**Client Application** is utilized to interact with the blockchain network by using RESTful APIs which provide the CTI sharing service to the organizational users. Application user interface is shown in Figure 3.2.



Cyber Threat Intelligence Sh

Sharing network is comprised with Hyperledger Fabric components.

Hyperledger Fabric is a private permissioned blockchain technology.

Figure 3.2: User Interface of Client Application

### 3.2.3 Application Scenario

After starting to listen the specified port, HTTP server is ready to receive requests from organizational entities. As a first step, the HTTP server needs to be connected to the CA server for admin identity enrollment and for user registration using the admin identity. Second, the registered user specifies the unique channel name and smart contract name using Fabric Node.js SDK and initiates the particular smart contract on the desired channel. While interacting with smart contracts, appropriate smart contract methods must be provided as arguments to the transaction functions for querying and updating the ledger state. Furthermore, during all these interactions with the chaincode, registered user identity is used for the signee of the requests. Last but not least, all ledger data requests are responded in JSON format.

The HTTP server application scenario is summarized step by step below;

- Enroll the admin identity to the Fabric CA.
- Register a user with the enrolled admin identity.
- Registered user invokes specified smart contract to query or update ledger state.
- Smart contract returns data to client applications in JSON format.

### 3.2.4 Chaincode

Chaincode forms the heart of a blockchain system as it provides a way to define the business logic that produces new data. In this proposed model chaincode is written in Node.js and deployed with different network configurations. As seen in Appendix A, CyberEvent class which extends Contract class is created and following methods are implemented in this smart contract.

- **createCyberEvent** method creates a cyber event from provided object. In this method we can check different tags in cyber event in order to decide to store and share in the blockchain network. For example, if the received MISP event object has TLP tag which restricts sharing sensitive information, chaincode checks this tag and executes the appropriate steps.
- **queryCyberEvent** method queries a single cyber event. It can also check the restriction tags while evaluating the query transaction.
- **queryCyberEventRange** method queries multiple cyber events within provided range ids.

### 3.2.5 Data Set

In this proposed solution, open source intelligence data feeds are stored and shared via blockchain network [102, 141]. These data sets are structured in MISP data format and suitable for integrating other security devices and sharing platforms. Sample MISP event in JSON format is presented in Appendix B.

# CHAPTER 4

# RESULTS

This chapter presents the tools and metrics used to evaluate the proposed blockchain network. Consequently, experiment results are discussed.

## 4.1 Experiment Setup

### 4.1.1 Metrics

Transaction throughput and latency is used for assessing the performance of our blockchain network. In blockchain benchmark tests, throughput is measured as successful transactions per second, whilst latency is the average response time of the requests made to the client application [142, 143]. While average response time is provided directly from our test tool, throughput is obtained with a simple calculation. Total failed requests are subtracted from total requests, and the resulted value is divided by test duration to find the transaction throughput as shown in above formula;

$$Throughput = \frac{\sum requests - \sum failures}{\sum duration} \tag{4.1}$$

### 4.1.2 Open Source Testing Tool

We used an open source testing tool, Locust, to evaluate throughput and transaction latency in our experiments. During a test, a web application is attacked by a swarm of users whose behavior is defined by Python code [144].

In our performance test scenario, users are attacking two specific interfaces, query and create cyber event, of the client application to measure the query and update metrics of the distributed ledgers. A simple locust scenario is created in Python for each test case as shown in Listing 4.1. The script was set up so that the simulated users made a request every second, and this allowed us to calculate the sending rate of our experiments.

**Listing 4.1** Locust test scenario

```python
import random
from locust import HttpUser, task, constant_pacing

class WebsiteUser(HttpUser):
    wait_time = constant_pacing(1)
    @task
    def add_event(self):
        self.client.get("/event/createevent")
```

### 4.1.3 Test Configurations

Test configurations are presented for performance and functionality tests in the following sections.

### 4.1.4 Performance Test Configurations

We have changed parameters and configurations in our performance experiments as shown in Table 4.1. Sequentially, we have changed the payload size, client application server processing mode, state database, block size, message count and channel population. Throughout the tests, when we encountered a configuration setting that improves performance, we continued the subsequent tests with this setting.

We have focused on create operations in assessments due to the fact that some of the configuration settings (block size, message count, channel population) only affect block generation. Evaluation of query operation has been conducted with parameters which gave best performance results. Architecture of the blockchain network for 2 organizations is depicted in Figure 4.1

Table 4.1: Test Configurations for Performance Evaluation

| Test Case | Payload Size (KB) | Client App Server | StateDB | Block Size (KB) | Message Count | Channel Population |
|---|---|---|---|---|---|---|
| 1 | 1, 10, 100 | Single Thread | CouchDB | 512 | 10 | 2 Orgs 4 Peers |
| 2 | 1 | Single Thread, Clustering | CouchDB | 512 | 10 | 2 Orgs 4 Peers |
| 3 | 1 | Clustering | CouchDB, LevelDB | 512 | 10 | 2 Orgs 4 Peers |
| 4 | 1 | Clustering | LevelDB | 8, 64, 512, 4096 | 10 | 2 Orgs 4 Peers |
| 5 | 1 | Clustering | LevelDB | 4096 | 2, 10, 50, 250 | 2 Orgs 4 Peers |
| 6 | 1 | Clustering | LevelDB | 4096 | 10 | 2 Orgs 4 Peers, 4 Orgs 8 Peers |

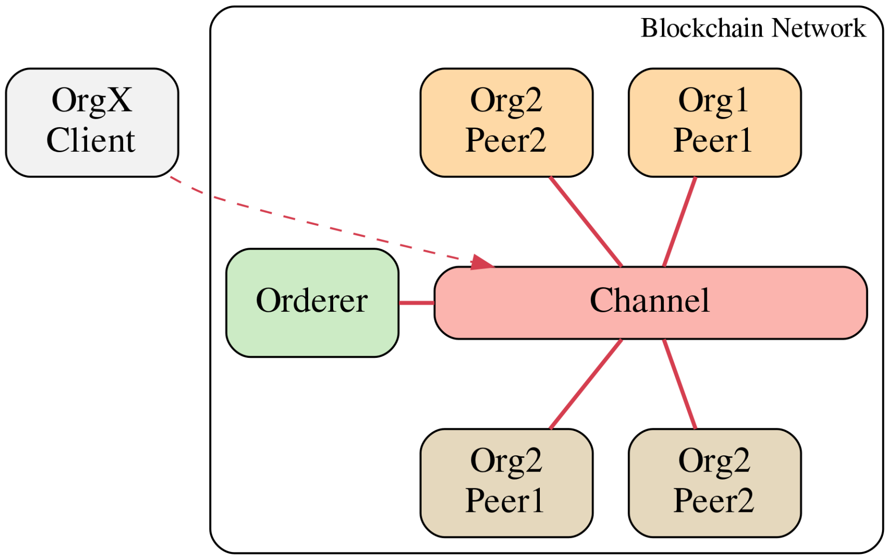


Figure 4.1: Blockchain Network Architecture for Performance Test

### 4.1.5 Functionality Test Configuration

We have performed functionality experiments to verify the private and automated secure sharing features of the Hyperledger Fabric consecutively. Two channels have been created and four organizations have participated in those channels for each test. Architecture of the blockchain network is depicted in Figure 4.2 In the first test, only one organization Org1 joined both channels, so we expected only Org1 would be able to access the assets of both channels.

In the second test, same architecture was employed with a different scenario. Two different chaincodes were installed to these two channels. They have different business logic regarding the TLP codes of shared events. One channel does not allow TLP amber labeled data and only allows TLP green and white labeled data, the other channel allows all TLP color labels except red. Therefore, we expected to control secure data sharing via chaincodes on those channels.

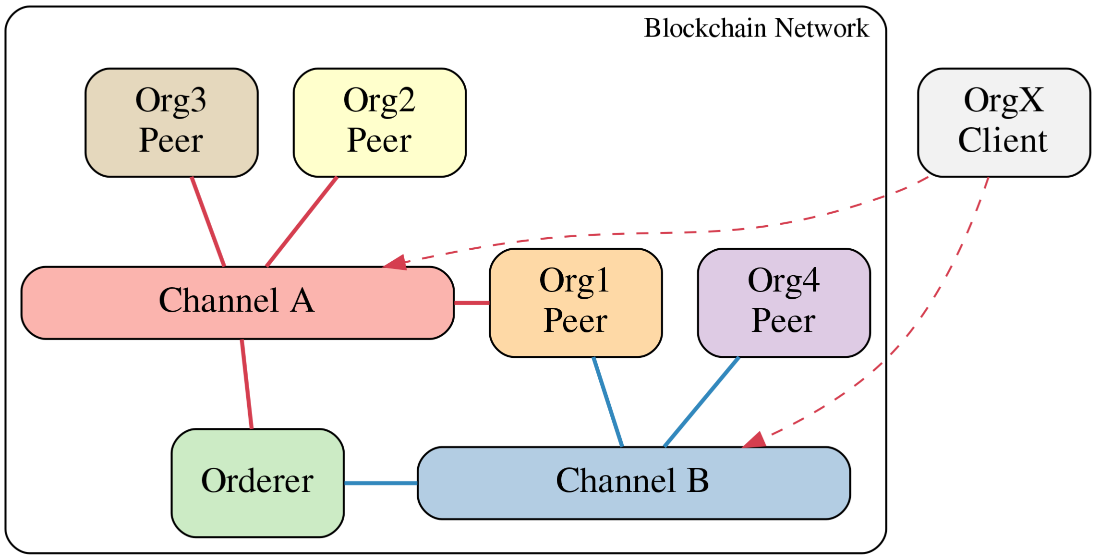


Figure 4.2: Blockchain Network Architecture for Functionality Test

## 4.2 Discussing Findings

Experiment results are discussed separately for performance and functionality tests.

### 4.2.1 Performance Test Findings

Performance test results are presented according to the metrics and test cases mentioned above.

#### 4.2.1.1 Payload Size

As seen in Figure 4.3, increase in the payload size decreases the create transaction throughput as we expected. For all payload sizes, the throughput increases linearly with the increase of send rate. The saturation point is at 40 transaction per second (TPS) for 1 KB payload, and no substantial increase is observed due to the latency increase shown in Figure 4.4. Measured metrics in this test case are shown in table 4.2

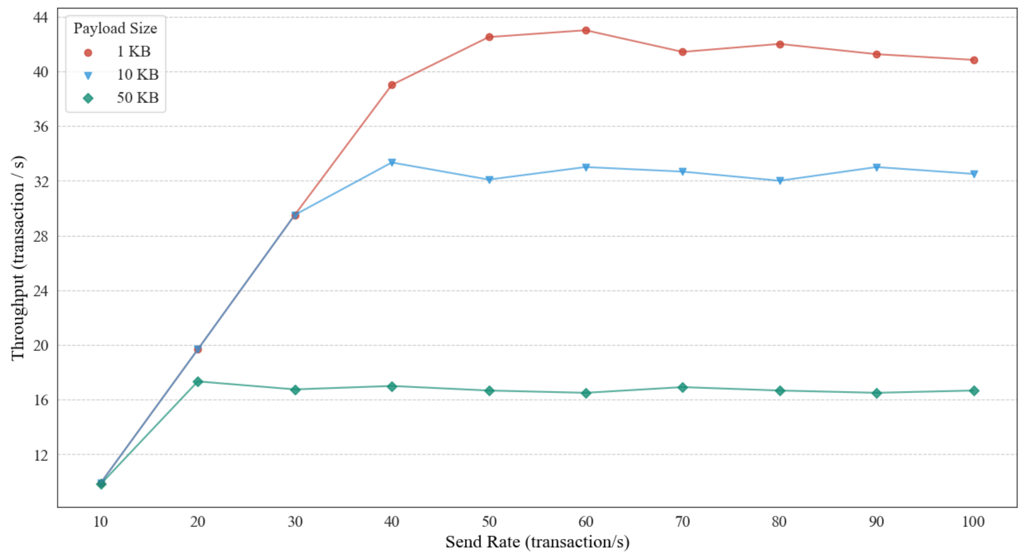


Figure 4.3: Throughput of Create Transactions for Different Payload Size

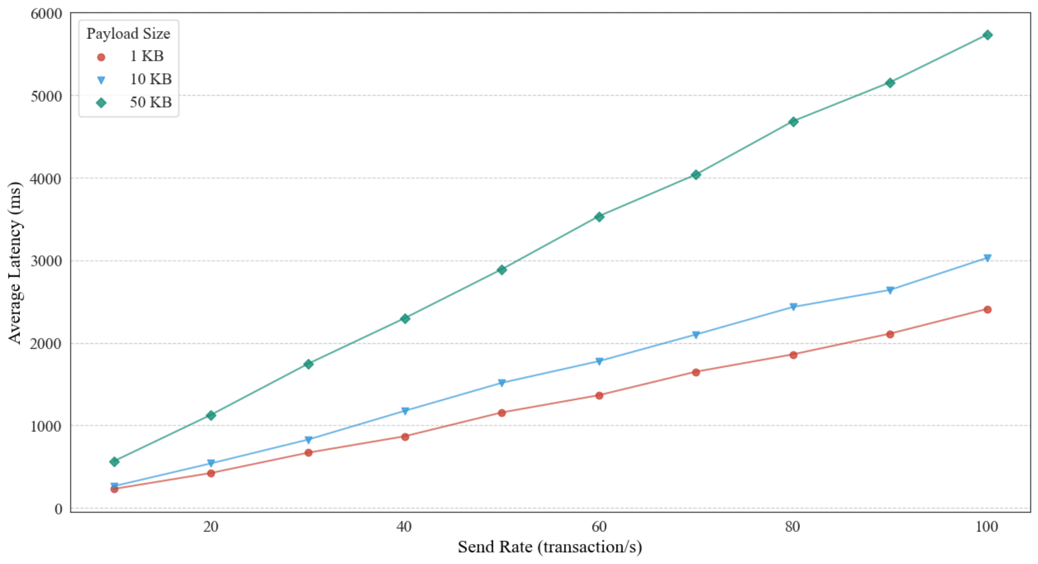


Figure 4.4: Latency of Create Transactions for Different Payload Size

#### 4.2.1.2 Client Application Server

One of the main performance bottlenecks is the single-threaded client application server. In order to overcome this hindrance and take advantage of our multi-core systems, a module named cluster is provided by Node.js. With the help of this module, we can initiate a cluster of child processes that all share the server ports. We have

Table 4.2: Test Results for Different Payload Size

| Transaction Type | Metric | | Payload Size | | |
|---|---|---|---|---|---|
| | | | 1 Kb | 10 Kb | 50 Kb |
| Create | Throughput (TPS) | mean | 34.9 | 28.8 | 16.1 |
| | | min | 9.9 | 9.9 | 9.8 |
| | | median | 41.0 | 32.3 | 16.7 |
| | | max | 43.0 | 33.3 | 17.3 |
| | Average Latency (ms) | mean | 1274.7 | 1630.6 | 3177.6 |
| | | min | 230.4 | 263.7 | 566.2 |
| | | median | 1261.6 | 1647.0 | 3213.2 |
| | | max | 2410.1 | 3028.7 | 5732.3 |

changed a little portion of our client application code to use cluster module as seen in Listing 4.2.

**Listing 4.2** Usage of Node.js cluster module

```
const cluster = require('cluster');
const express = require('express');
const numCPUs = require('os').cpus().length;
if (cluster.isMaster) {
  for (var i = 0; i < numCPUs; i++) {
    // create a worker for each core
    cluster.fork();
  }
} else {
  // workers share the TCP connection in this server
  // all workers use this port
  app.listen(8080);
}
```

If we inspect the test results in Figures 4.5, 4.6 and Table 4.3, we can see the performance difference between processing modes. It is obvious that clustering mode increased the throughput and decreased latency almost 25%.

#### 4.2.1.3 State Database

Hyperledger Fabric allows to use two different peer state database options; CouchDB and LevelDB. Although CouchDB supports rich queries of JSON data objects it is

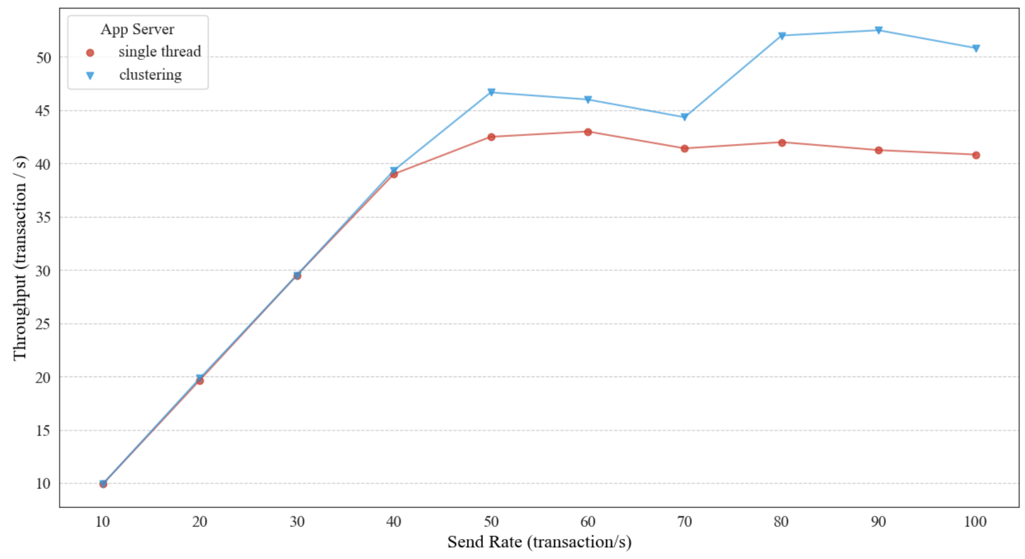


Figure 4.5: Throughput of Create Transactions for Different Server Processing Modes

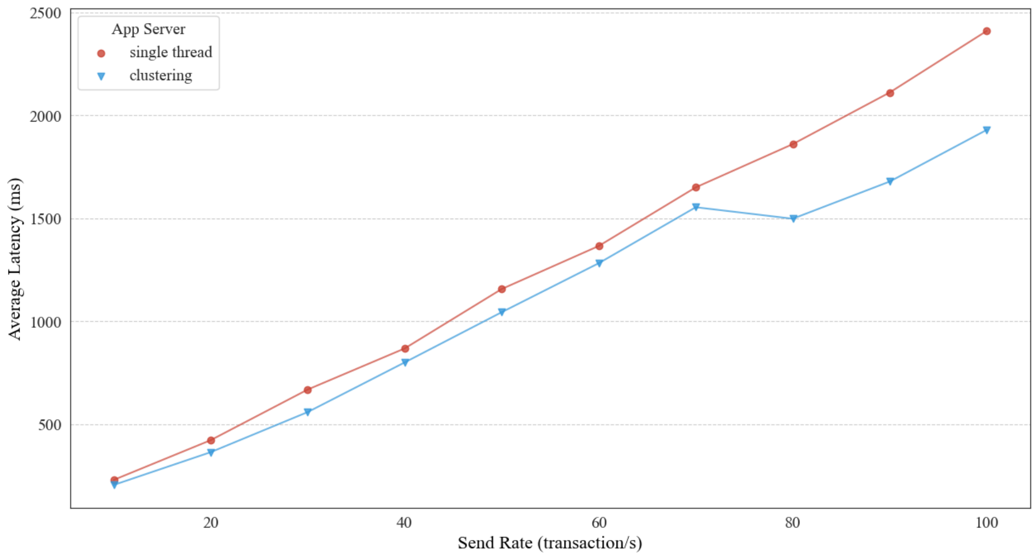


Figure 4.6: Latency of Create Transactions for Different Server Processing Modes

slower than LevelDB option. The reason for performance difference between these two options is that the former is an external database and can be accessed through REST API while the latter is embedded within peers [142]. After all, we observed that LevelDB outperforms CouchDB with 75% increase in maximum transaction throughput and 45% decrease in average latency as shown in Figures 4.7, 4.8 and Table 4.4.

Table 4.3: Test Results for Different Server Processing Modes

| Transaction Type | Metric | | Client App Server | |
|---|---|---|---|---|
| | | | Single Thread | Clustering |
| Create | Throughput (TPS) | mean | 34.9 | 39.1 |
| | | min | 9.9 | 9.9 |
| | | median | 41.0 | 45.2 |
| | | max | 43.0 | 52.5 |
| | Average Latency (ms) | mean | 1274.7 | 1092.0 |
| | | min | 230.4 | 205.9 |
| | | median | 1261.6 | 1163.5 |
| | | max | 2410.1 | 1929.7 |

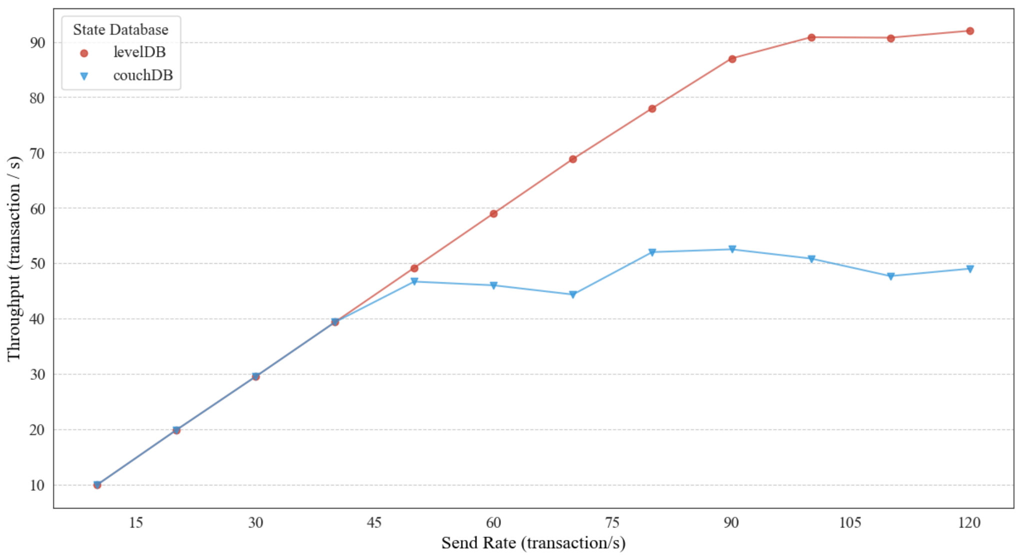


Figure 4.7: Throughput of Create Transactions for Different Database Options

#### 4.2.1.4 Block Size

There are several strategies for batching the blocks in Fabric. These strategies determine the transaction count or transaction size in bytes to be collected by orderer before batching a block. These are set in the `configtx.yaml` file as seen in Listing 4.3. The throughput is affected by the configuration of these parameters.

`PreferredMaxBytes` represents the block size in bytes and `MessageCount` represents the message count. While block size sets the size limit, message count sets the count

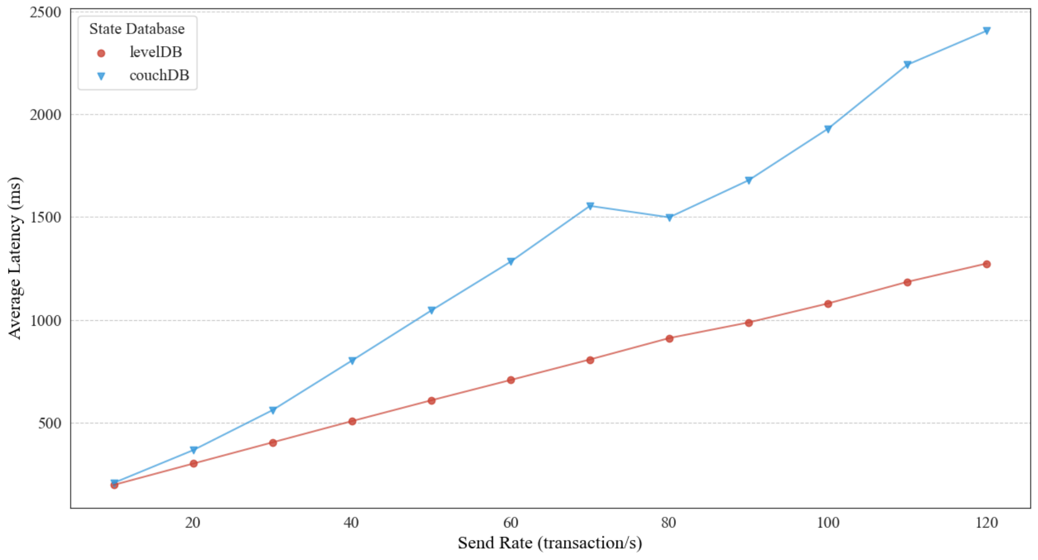


Figure 4.8: Latency of Create Transactions for Different Database Options

Table 4.4: Test Results for Different Database Options

| Transaction Type | Metric | | State Database | |
|---|---|---|---|---|
| | | | CouchDB | LevelDB |
| Create | Throughput (TPS) | mean | 40.6 | 59.5 |
| | | min | 9.9 | 9.9 |
| | | median | 46.3 | 63.9 |
| | | max | 52.5 | 92.0 |
| | Average Latency (ms) | mean | 1297.3 | 746.8 |
| | | min | 205.9 | 196.1 |
| | | median | 1390.3 | 756.6 |
| | | max | 2406.5 | 1273.8 |

limit of the transactions. The orderer groups transactions into blocks in accordance with these limits. We have tested block size effect in this section and message count effect in the following section.

Previous tests on Fabric show that when a high transaction influx is expected, a higher block size increases transaction throughput and decreases latency [142]. However, there is optimal value for each system under test [143]. In our tests, we observed that throughput increased and latency decreased with the increase of block size as expected. As a result 4096 KB is the optimum value for our network (see Figures 4.9, 4.10 and Table 4.5).

**Listing 4.3** Orderer section of configtx.yaml

```
Orderer: &OrdererDefaults
    OrdererType: solo
    Addresses:
        - orderer.cti.com.tr:7050
    BatchTimeout: 2s
    BatchSize:
        MaxMessageCount: 10
        AbsoluteMaxBytes: 99 MB
        PreferredMaxBytes: 4096 KB
```

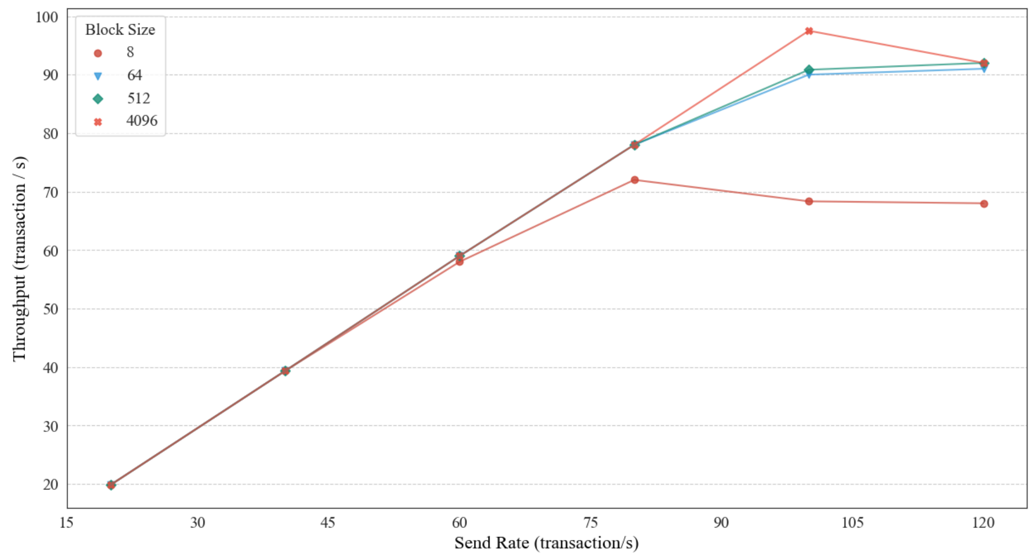


Figure 4.9: Throughput of Create Transactions for Different Block Size

#### 4.2.1.5 Message Count

In this section, we have assessed our network performance for different message count settings. As Figures 4.11, 4.12 and Table 4.6 indicates, throughput increased and latency decreased with the increase of message count for up to the value of 10. After that point, throughput decreased and latency increased with the increase of message count.

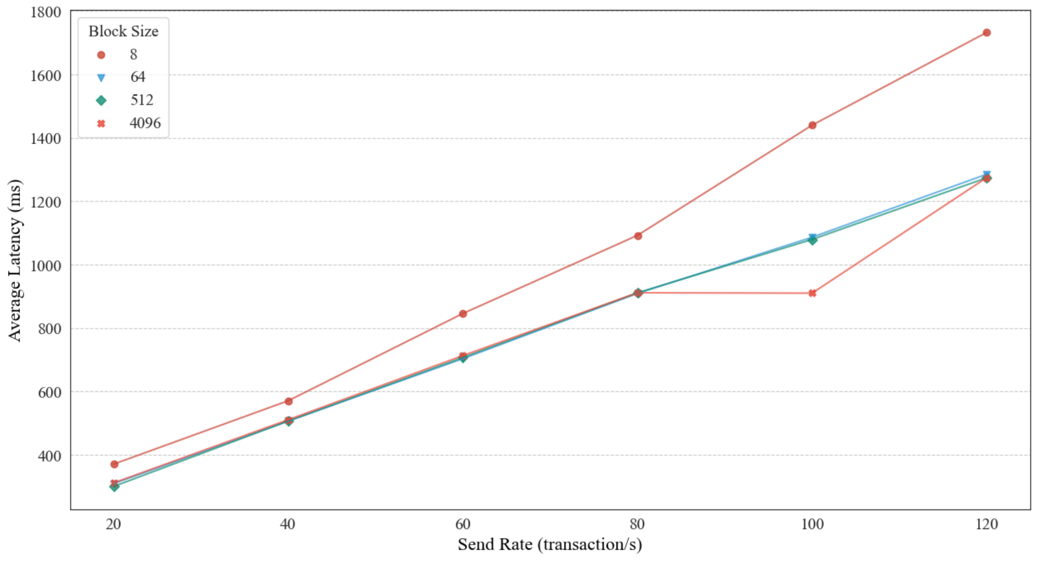


Figure 4.10: Latency of Create Transactions for Different Block Size

Table 4.5: Test Results for Different Block Size

| Transaction Type | Metric | | Block Size (KB) | | | |
|---|---|---|---|---|---|---|
| | | | 8 | 64 | 512 | 4096 |
| Create | Throughput (TPS) | mean | 54.2 | 62.9 | 63.2 | 64.3 |
| | | min | 19.8 | 19.8 | 19.8 | 19.8 |
| | | median | 63.0 | 68.5 | 68.5 | 68.5 |
| | | max | 72.0 | 91.0 | 92.0 | 97.5 |
| | Average Latency (ms) | mean | 1008.5 | 799.7 | 796.1 | 771.5 |
| | | min | 370.3 | 308.8 | 299.8 | 311.0 |
| | | median | 968.9 | 806.0 | 808.7 | 811.2 |
| | | max | 1732.5 | 1285.1 | 1273.8 | 1274.4 |

#### 4.2.1.6 Channel Population

Another aspect that decreases the throughput and increases the latency is channel population due to the fact that our endorsement policy uses *and* logic which means at least one peer for each organization must endorse our transaction proposals (see Listing 4.4).

As compared in Figures 4.13, 4.14 and Table 4.7, throughput decreases and latency increases with the increase in channel population.

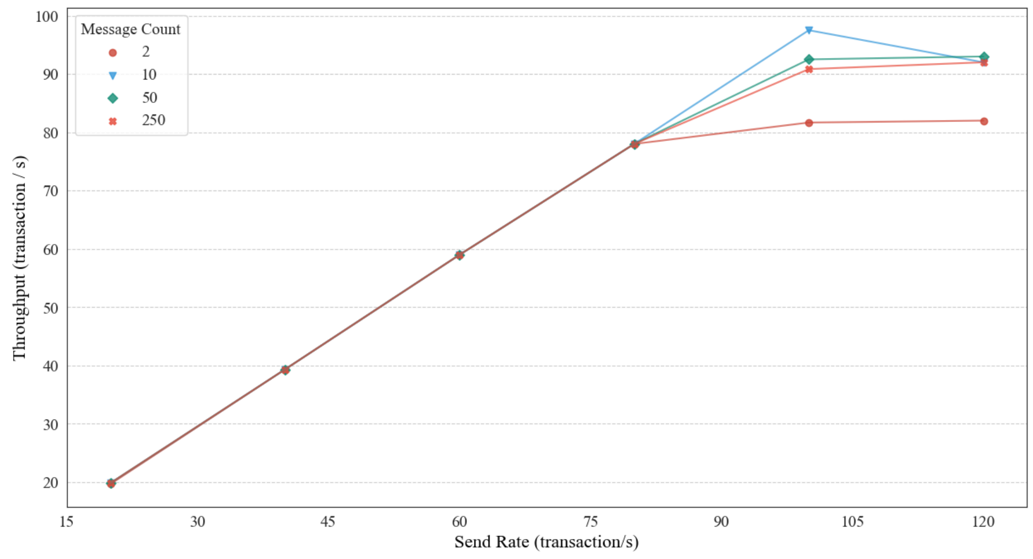


Figure 4.11: Throughput of Create Transactions for Different Message Count

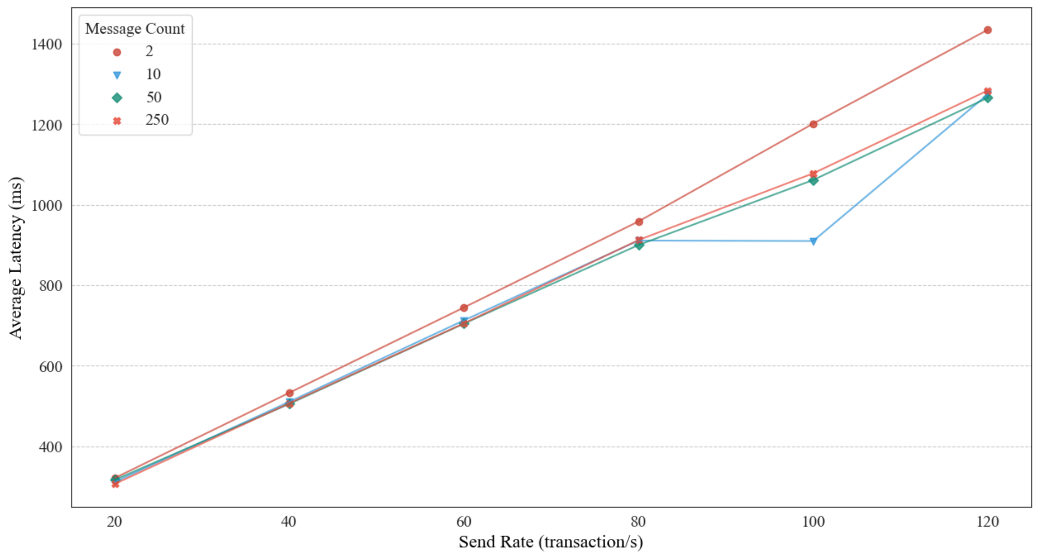


Figure 4.12: Latency of Create Transactions for Different Message Count

#### 4.2.1.7 Query Operations

Create operations are more time consuming than query operations due to the consensus mechanism. While query process requires only one peer, commit process involves multiple peer endorsements and orderer. Thus, we have observed high throughput and low latency while querying the ledger in our experiments.

Table 4.6: Test Results for Different Message Count

| Transaction Type | Metric | | Message Count | | | |
|---|---|---|---|---|---|---|
| | | | 2 | 10 | 50 | 250 |
| Create | Throughput (TPS) | mean | 59.9 | 64.3 | 63.6 | 63.2 |
| | | min | 19.7 | 19.8 | 19.8 | 19.8 |
| | | median | 68.5 | 68.5 | 68.5 | 68.5 |
| | | max | 82.0 | 97.5 | 93.0 | 92.0 |
| | Average Latency (ms) | mean | 865.5 | 771.5 | 792.2 | 798.7 |
| | | min | 320.9 | 311.0 | 316.4 | 306.3 |
| | | median | 851.6 | 811.2 | 802.2 | 808.7 |
| | | max | 1434.6 | 1274.4 | 1265.6 | 1283.7 |

Table 4.7: Test Results for Different Channel Population

| Transaction Type | Metric | | Channel Population | |
|---|---|---|---|---|
| | | | 2 Orgs | 4 Orgs |
| Create | Throughput (TPS) | mean | 64.3 | 51.3 |
| | | min | 19.8 | 19.7 |
| | | median | 68.5 | 60.8 |
| | | max | 97.5 | 64.0 |
| | Average Latency (ms) | mean | 771.5 | 1100.5 |
| | | min | 311.0 | 385.0 |
| | | median | 811.2 | 1084.9 |
| | | max | 1274.4 | 1850.0 |

**Listing 4.4** Chaincode instantiation with 4 organizations

```
# instantiate the chaincode
ORDERER_ADDRESS="orderer.cti.com.tr:7050"
CHANNEL_NAME="channelcti"
CHAINCODE_NAME="CyberEvent"
VERSION="1.0"
LANGUAGE="node"
peer chaincode instantiate -o $ORDERER_ADDRESS -C $CHANNEL_NAME \
                    -n $CHAINCODE_NAME -v $VERSION -l $LANGUAGE \
                    -c '{"Args":["instantiate"]}' \
                    -P "AND ('Org1MSP.peer','Org2MSP.peer', \
                         'Org3MSP.peer','Org4MSP.peer')"
```

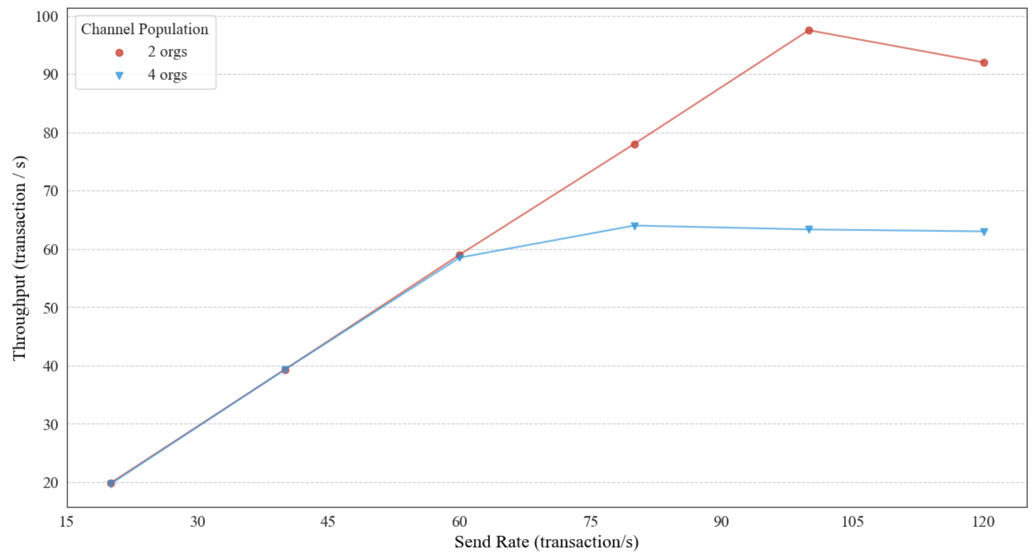


Figure 4.13: Throughput of Create Transactions for Different Channel Population

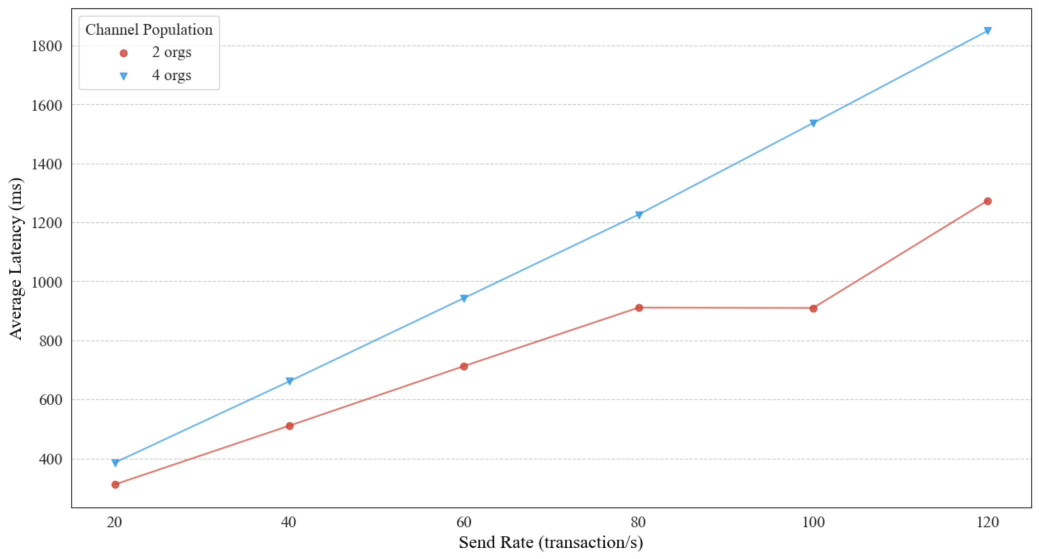


Figure 4.14: Latency of Create Transactions for Different Channel Population

Figures 4.15, 4.16 illustrate the throughput and latency of query interface. They indicate that throughput increases linearly with low latency until it reaches the maximum level of 240 TPS as seen in Table 4.8.

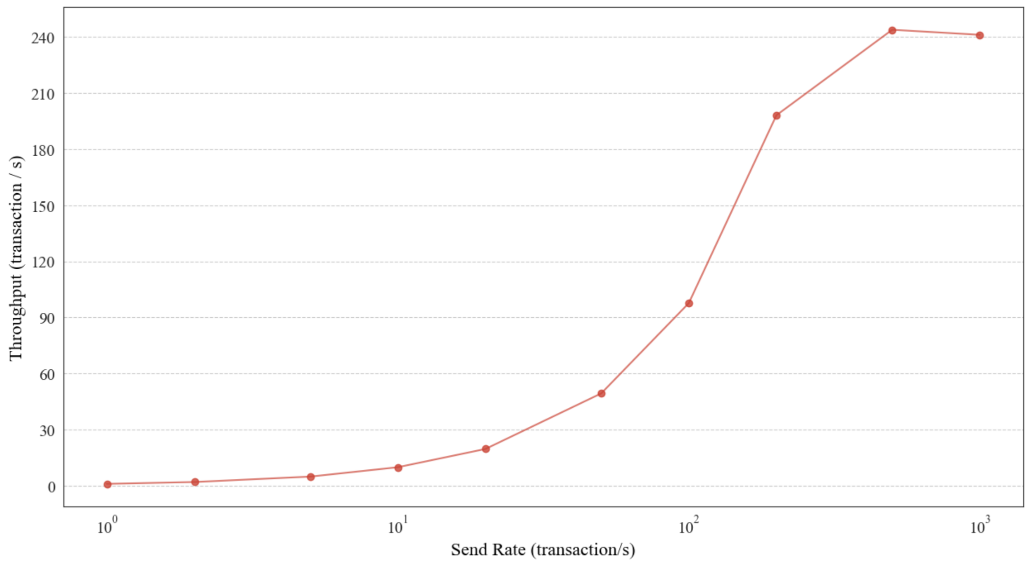


Figure 4.15: Throughput of Query Transactions

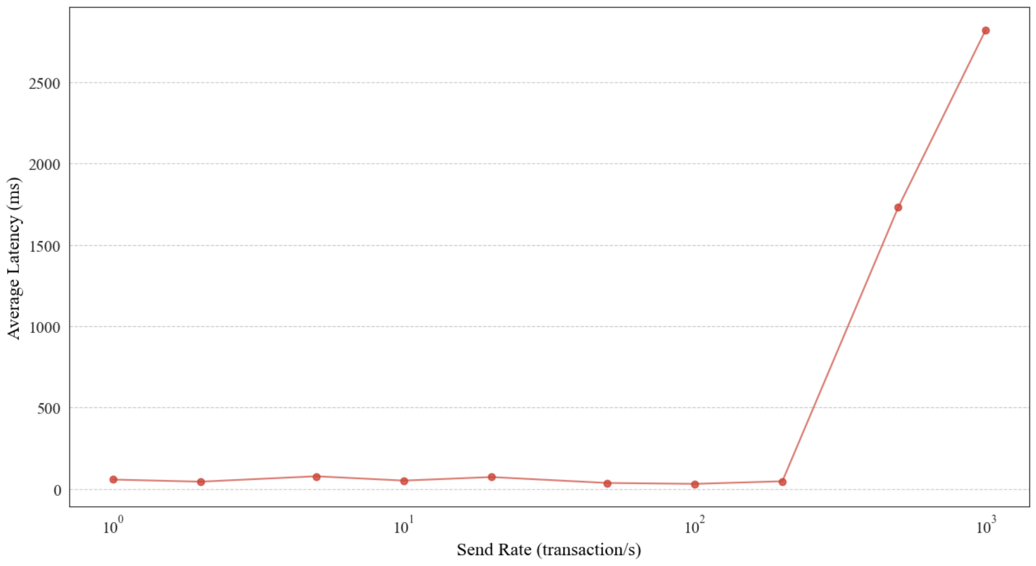


Figure 4.16: Latency of Query Transactions

#### 4.2.1.8 Comparison

Fabric performance depends on many conditions, like Fabric configuration parameters, smart contract code complexity, endorsement policy, underlying infrastructure resources(processor, memory, storage, etc.). However, we compared the performance results with other studies.

Table 4.8: Test Results for Query Transactions

| Transaction Type | Metric | | |
|---|---|---|---|
| Query | Throughput (TPS) | mean | 86.7 |
| | | min | 1.0 |
| | | median | 34.5 |
| | | max | 243.8 |
| | Average Latency (ms) | mean | 496.8 |
| | | min | 30.8 |
| | | median | 54.1 |
| | | max | 2822.2 |

Collaborative DDoS attack mitigation system based on Hyperledger Fabric is presented in [133] and throughput results are shown in Figure 4.17. There is not enough information about the Fabric configuration in this study but the results are similar to ours.

Another results that we compared is the official performance report of Hyperledger Fabric [145]. As it is seen in Figure 4.18, throughput is greater than our results. It can be due to the fact that they have used more decent processor, memory and storage resources. But the pattern of changes in throughput is same as our experiments. Throughput is highly affected by the payload size and database choice.

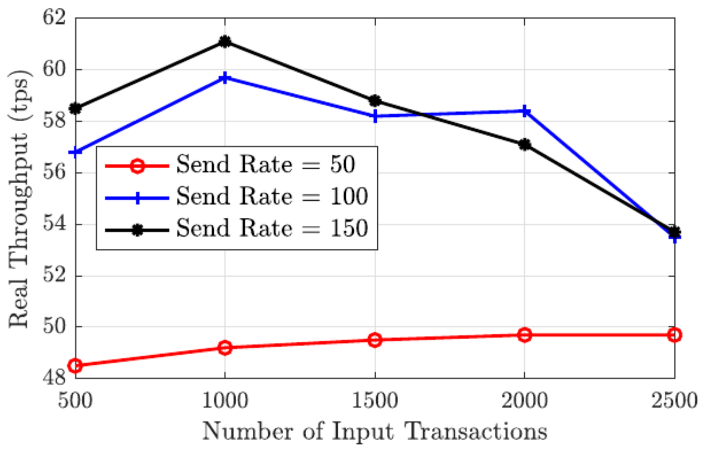


Figure 4.17: Throughput Results for Comparison Retrieved from [133]

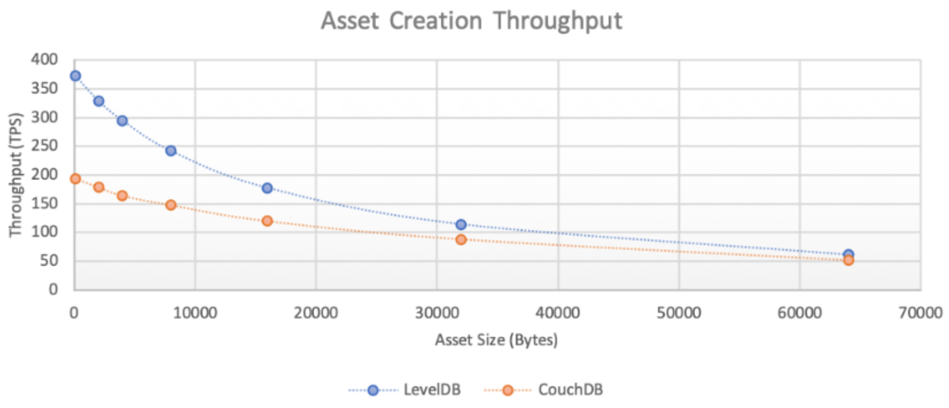


Figure 4.18: Throughput Results for Comparison
Retrieved from [145]

### 4.2.2 Functionality Test Findings

Firstly, for evaluating the privacy functionality, create and query transactions have been performed with different organizational ids and access permissions to the channels, chaincodes, and ledgers have been observed. Since it is a member of both channels, we have observed that only Org1 can access both channels. Furthermore, we have confirmed that other organizations can only access the channel of which it is a member. Test results are summarized in Table 4.9

As a result of this test, we have verified that our proposed solution allows different sectors to share information over different channels in the same network. Sectoral CERTs will have the ability to share threats targeting their specific sectors through such private channels. This is an important feature in ensuring the security of sensitive data.

In the second functionality test, different chaincodes have been instantiated on channels. Chaincode on channel A allows TLP amber and above labels which we expected to share all TLP labeled data except red. However, chaincode on channel B allows only TLP green and white labeled data. With those different chaincodes, we have observed that we cannot share unallowed labels on the channels. Results are shown in Table 4.10.

Table 4.9: Test Results for Privacy Functionality

| Organization ID | Test Case | | Access result |
|---|---|---|---|
| Org1 | Channel A | Query | + |
| | | Create | + |
| | Channel B | Query | + |
| | | Create | + |
| Org2 | Channel A | Query | + |
| | | Create | + |
| | Channel B | Query | Denied |
| | | Create | Denied |
| Org3 | Channel A | Query | + |
| | | Create | + |
| | Channel B | Query | Denied |
| | | Create | Denied |
| Org4 | Channel A | Query | Denied |
| | | Create | Denied |
| | Channel B | Query | + |
| | | Create | + |

Accordingly, we have verified that we can automate controlling the sharing policy labels and implement critical business logic with chaincodes in the blockchain. This feature improves the efficiency of secure sharing and enhances the timeliness of the data.

Table 4.10: Test Results for Automated Secure Sharing Functionality

| Organization ID | Test Case | | Sharing result |
|---|---|---|---|
| Org1 | Channel A | TLP Red | Denied |
| | | TLP Amber | + |
| | | TLP Green | + |
| | | TLP White | + |
| | Channel B | TLP Red | Denied |
| | | TLP Amber | Denied |
| | | TLP Green | + |
| | | TLP White | + |
| Org2 | Channel A | TLP Red | Denied |
| | | TLP Amber | + |
| | | TLP Green | + |
| | | TLP White | + |
| Org3 | Channel A | TLP Red | Denied |
| | | TLP Amber | + |
| | | TLP Green | + |
| | | TLP White | + |
| Org4 | Channel B | TLP Red | Denied |
| | | TLP Amber | Denied |
| | | TLP Green | + |
| | | TLP White | + |

# CHAPTER 5

# CONCLUSION

In this chapter we summarize the effort and contributions and present potential future extensions of this work.

## 5.1 Contribution and Discussion

Even if an organization has achieved cyber security today, it cannot be guaranteed that it will not be exposed to cyber security threats in the future. Cyber Threat Intelligence (CTI) is the most valuable ability to detect, prevent, respond to today's sophisticated threats in a timely and effective manner. The value of this skill depends on the quality of the data collected and shared. Thus, the quality of the data is highly related to those features which are reliability, accuracy, relevance, interoperability and, most importantly, timely readiness. It is observed that the product offered by open source and commercial providers often does not meet these features. This causes users to not trust sharing platforms/communities and avoid cyber-threat intelligence exchange. Another reason that interferes with data sharing is that the sharing standards are very diverse and many stakeholders present data at different standards.

In this study, we explained CTI thoroughly. Afterwards, the most relevant and contemporary reporting formats for CTI sharing are briefly described and compared according to their support of other data exchange standards. In this regard, STIX and MISP widely support the other standards and as a consequence they are mostly adopted by threat intelligence sharing tools. Moreover we present a new CTI exchange model based on Blockchain and evaluated the characteristics of our model through experimentation. In order to analyze the effectiveness of our solution, we implemented different blockchain configurations which has separate security mechanisms provided by Hyperledger Fabric. We measured the performance of the network in different conditions to see the effects of the configuration changes. Furthermore, we divided our CTI network into different channels to share sensitive sectoral threat data and examined the privacy aspect of our model. Finally, we also evaluated the efficacy of using chaincodes in order to implement critical business logic like executing a sharing policy. These assessments helped us not only to verify the advantages but also to identify possible constraints, regarding the storage, querying, processing, security and privacy of transactions.

As a result, with this offering a suitable solution is provided to some of the open challenges described in Section 2.12. See Table 2.4 and Table 2.5.

## 5.2 Limitations and Future Work

On the other hand, we identified some limitations and future research directions during the assessment of our implementation.

There are several security limitations intrinsic to the blockchain technology itself. For these constraints, some workarounds could be implemented. Malicious nodes are not considered so subsequent work could include malicious effects on our model. To provide a production grade solution more secure consensus protocol different than Solo may be used. We evaluated our solution with limited number of nodes due to the computational constraints, therefore scalability of the model couldn't be tested. Tests which simulate practical use of the network in real world situations can be performed. Our chaincode has limited capabilities at some extent. A chaincode which allows us to ingest various data formats can be developed in the future. Inclusion of these issues in future studies should be considered.

# REFERENCES

[1] "Internet Live Stats," 2019. [Online]. Available: http://www.internetlivestats.com/. [Accessed: 02-Dec-2019].

[2] Schneier, B., "Software Complexity and Security," 2000. [Online]. Available: https://www.schneier.com/crypto-gram/archives/2000/0315.html#8. [Accessed: 02-Dec-2018].

[3] Hern, A. and Gibbs, S., "What is WannaCry ransomware and why is it attacking global computers?," *The Guardian*, vol. 12, 2017.

[4] Thomson, I., "Everything you need to know about the petya, er, notpetya nasty trashing pcs worldwide," *The Register*, 2017.

[5] "World Economic Forum Global Risks Report 2019," 2019. [Online]. Available: http://www3.weforum.org/docs/WEF_Global_Risks_Report_2019.pdf. [Accessed: 10-Dec-2019].

[6] "World's biggest data breaches," 2019. [Online]. Available: http://www.informationisbeautiful.net/visualizations/worlds-biggest-data-breaches-hacks/. [Accessed: 02-Dec-2019].

[7] Symantec,, "Internet Security Threat Report 2019," 2019. [Online]. Available: https://www.symantec.com/content/dam/symantec/docs/reports/istr-24-2019-en.pdf. [Accessed: 21-Dec-2019].

[8] Zibak, A. and Simpson, A., "Cyber threat information sharing: Perceived benefits and barriers," in *Proceedings of the 14th International Conference on Availability, Reliability and Security*, p. 85, ACM, 2019.

[9] Watanabe, F., "Fifteen axioms for intelligence analysts," tech. rep., Central Intelligence Agency Washington DC Center for the Study of Intelligence, 1997.

[10] Cohen, I. S., *Realpolitik, Theory and Practice*. Dickenson Pub. Co., 1975.

[11] "Merriam-Webster Definiton of Intelligence," 2020. [Online]. Available: https://www.merriam-webster.com/dictionary/intelligence. [Accessed: 08-Jan-2020].

[12] Walsh, P. F., *Intelligence and Intelligence Analysis*. Willan, 2011.

[13] Prunckun, H., *Handbook of Scientific Methods of Inquiry for Intelligence Analysis*, vol. 11. Scarecrow Press, 2010.

[14] Whitaker, R., *The End of Privacy: How Total Surveillance is Becoming a Reality*. Scribe Publications, 2000.

[15] Mcmillan, R., "Definition: Threat intelligence," *Gartner*, 2013.

[16] "SANS Threat Intelligence Definiton," 2020. [Online]. Available: https://www.sans.org/course/cyber-threat-intelligence. [Accessed: 08-Jan-2020].

[17] Ratcliffe, J. H., *Intelligence-led policing*. Routledge, 2016.

[18] Steele, R. D., "Open source intelligence," *Handbook of intelligence studies*, vol. 42, no. 5, pp. 129–147, 2007.

[19] Chismon, D. and Ruks, M., "Threat intelligence: Collecting, analysing, evaluating," *MWR InfoSecurity Ltd*, 2015.

[20] "Russia accused of unleashing cyberwar to disable Estonia," 2007. [Online]. Available: https://www.theguardian.com/world/2007/may/17/topstories3.russia. [Accessed: 16-Jan-2020].

[21] "Cooperative Cyber Defence Centre of Excellence (CCDCOE)," 2020. [Online]. Available: https://ccdcoe.org/about-us/. [Accessed: 16-Jan-2020].

[22] Bundestag, D., "Gesetz zur Erhöhung der Sicherheit informationstechnischer Systeme (IT-Sicherheitsgesetz)," *Bundesgesetzblatt, I (31)*, pp. 1324–1331, 2015.

[23] Directive, N., "Directive (EU) 2016/1148 of the European Parliament and of the Council of 6 July 2016 concerning measures for a high common level of security of network and information systems across the Union," 2016. [Online]. Available: https://publications.europa.eu/en/publication-detail/-/publication/d2912aca-4d75-11e6-89bd-01aa75ed71a1/language-en. [Accessed: 16-Jan-2020].

[24] "NATO and the European Union enhance cyber defence cooperation," 2016. [Online]. Available: https://www.nato.int/cps/en/natohq/news_127836.htm. [Accessed: 16-Jan-2020].

[25] Order, E., "Promoting Private Sector Cybersecurity Information Sharing," 2015. [Online]. Available: https://www.federalregister.gov/documents/2015/02/20/2015-03714/promoting-private-sector-cybersecurity-information-sharing. [Accessed: 16-Jan-2020].

[26] Act,, "The Cybersecurity Information Sharing Act," 2014. [Online]. Available: https://www.congress.gov/bill/113th-congress/senate-bill/2588. [Accessed: 16-Jan-2020].

[27] "National Cybersecurity and Communicaitons Integration Center (NCCIC)," 2020. [Online]. Available: https://www.us-cert.gov/about-us. [Accessed: 16-Jan-2020].

[28] "Cyber Threat Intelligence Integration Center (CTIIC)," 2020. [Online]. Available: https://www.dni.gov/index.php/ctiic-home. [Accessed: 16-Jan-2020].

[29] "Turkish National CERT Communication Platform," 2018. [Online]. Available: https://www.btk.gov.tr/haberler/siber-guvenligimizi-artirma-azim-ve-kararliligi-icerisindeyiz. [Accessed: 16-Jan-2020].

[30] "News about the draft of cyber security act in Turkey," 2017. [Online]. Available: https://www.aa.com.tr/tr/politika/siber-guvenlik-kanun-taslagi-calismasini-tamamladik/984785. [Accessed: 16-Jan-2020].

[31] Martin, R. A., "Making security measurable and manageable," in *Military Communications Conference, 2008. MILCOM 2008. IEEE*, pp. 1–9, IEEE, 2008.

[32] Barnum, S., "Standardizing cyber threat intelligence information with the structured threat information expression (stix)," *MITRE Corporation*, vol. 11, pp. 1–22, 2012.

[33] "The Trusted Automated eXchange of Indicator Information (TAXII)," 2012. [Online]. Available: https://www.mitre.org/publications/technical-papers/the-trusted-automated-exchange-of-indicator-information-taxii%E2%84%A2. [Accessed: 11-Feb-2020].

[34] Stoll, C., *The Cuckoo's Egg: Tracking a spy through the maze of computer espionage*. Simon and Schuster, 2005.

[35] "Transition of STIX/TAXII to an International Standards Organization," 2015. [Online]. Available: http://making-security-measurable.1364806.n2.nabble.com/STIX-Transition-of-STIX-TAXII-to-an-International-Standards-Organization-td7586826.html. [Accessed: 17-Aug-2018].

[36] Danyliw, R., Meijer, J., and Demchenko, Y., "The incident object description exchange format," tech. rep., Internet Engineering Task Force (IETF), 2007.

[37] Moriarty, K., "Real-time inter-network defense (rid)," tech. rep., Internet Engineering Task Force (IETF), 2010.

[38] Cam-Winget, N., Appala, S., Pope, S., and Saint-Andre, P., "Using xmpp for security information exchange," tech. rep., Internet Engineering Task Force (IETF), 2018.

[39] "OpenIOC," 2013. [Online]. Available: https://www.fireeye.com/blog/threat-research/2013/10/openioc-basics.html. [Accessed: 11-Feb-2020].

[40] "AlienVault Open Threat Exchange (AV-OTX) released," 2012. [Online]. Available: https://www.alienvault.com/blogs/labs-research/alienvault-open-threat-exchange-av-otx-released. [Accessed: 11-Feb-2020].

[41] "Check Point launches ThreatCloud IntelliStore," 2014. [Online]. Available: https://www.checkpoint.com/press/2014/check-point-pioneers-revolutionary-cyber-intelligence-marketplace-threatcloud-intellistore/. [Accessed: 11-Feb-2020].

[42] "Facebook launches social network for security pros," 2015. [Online]. Available: https://www.cnet.com/news/facebook-launches-social-network-for-sharing-security-threat-info/. [Accessed: 11-Feb-2020].

[43] "IBM opens threat intelligence to combat cyber attacks," 2015. [Online]. Available: https://www-03.ibm.com/press/us/en/pressrelease/46634.wss. [Accessed: 11-Feb-2020].

[44] "Symantec launches cyberthreat intelligence service," 2015. [Online]. Available: https://www.zdnet.com/article/know-thy-enemy-symantec-launches-cyberthreat-intelligence-service-for-the-enterprise/. [Accessed: 11-Feb-2020].

[45] "FireEye Announces Acquisition of iSIGHT Partners," 2016. [Online]. Available: http://investors.fireeye.com/releasedetail.cfm?ReleaseID=951017. [Accessed: 11-Feb-2020].

[46] "NC4 to buy cyber threat intelligence company, Soltra," 2016. [Online]. Available: https://www.prnewswire.com/news-releases/nc4-to-buy-cyber-threat-intelligence-company-soltra-from-fs-isac-dtcc-300367984.html. [Accessed: 11-Feb-2020].

[47] "AT&T to Acquire AlienVault," 2018. [Online]. Available: https://www.alienvault.com/who-we-are/press-releases/at-t-to-acquire-alienvault. [Accessed: 11-Feb-2020].

[48] Dandurand, L., Kaplan, A., Kácha, P., Kadobayashi, Y., Kompanek, A., Millar, T., Nazario, J., Perlotto, R., and Young, W., *Standards and Tools for Exchange and Processing of Actionable Information*. European Network and Information Security Agency ENISA, 2015.

[49] "Tcpdump Project," 2019 (last released September 30, 2019). [Online]. Available: http://www.tcpdump.org/. [Accessed: 17-Mar-2020.

[50] "Wireshark," 2020. [Online]. Available: https://www.wireshark.org/. [Accessed: 17-Mar-2020].

[51] "Snort IDS/IPS," 2020. [Online]. Available: https://snort.org. [Accessed: 17-Mar-2020.

[52] "NetworkMiner," 2020. [Online]. Available: http://www.netresec.com/?page=NetworkMiner. [Accessed: 17-Mar-2020].

[53] "Libpcap in C and Python," 2020. [Online]. Available: http://www.tcpdump.org/ and https://pypi.org/project/libpcap/. [Accessed: 17-Mar-2020].

[54] Claise, B., "RFC 3954," *Cisco Systems Netflow Services Export Version*, vol. 9, 2004.

[55] "SiLK," 2020. [Online]. Available: https://tools.netsa.cert.org/silk/index.html. [Accessed: 17-Mar-2020].

[56] "Argus," 2020. [Online]. Available: http://qosient.com/argus/. [Accessed: 17-Mar-2020].

[57] "Ntopng," 2020. [Online]. Available: https://www.ntop.org/products/traffic-analysis/ntop/ and https://github.com/ntop/ntopng. [Accessed: 17-Mar-2020].

[58] Quittek, J., Zseby, T., Claise, B., and Zander, S., "RFC 3917: Requirements for IP Flow Information Export: IPFIX," *Published by Internet Engineering Task Force (IETF). Internet Society (ISOC) RFC Editor. USA.*, 2004.

[59] "LibIPFIX," 2013. [Online]. Available: https://sourceforge.net/projects/libipfix/. [Accessed: 17-Mar-2020].

[60] "CEF," 2016. [Online]. Available: https://www.secef.net/wp-content/uploads/sites/10/2017/04/CommonEventFormatv23.pdf. [Accessed: 17-Mar-2020.

[61] "Suricata IDS/IPS," 2020. [Online]. Available: https://suricata-ids.org/. [Accessed: 17-Mar-2020].

[62] "Zeek (Formerly Bro) IDS/IPS," 2020. [Online]. Available: https://www.zeek.org. [Accessed: 17-Mar-2020].

[63] "YARA," 2020. [Online]. Available: https://github.com/VirusTotal/yara and https://yara.readthedocs.io/en/v3.11.0/. [Accessed: 17-Mar-2020].

[64] "Who is using YARA?," 2020. [Online]. Available: https://github.com/VirusTotal/yara/blob/master/README.md#whos-using-yara. [Accessed: 17-Mar-2020].

[65] "OpenIOC," 2013. [Online]. Available: https://github.com/mandiant/OpenIOC_1.1. [Accessed: 20-Mar-2020].

[66] "IOC-writer," 2013. [Online]. Available: https://github.com/mandiant/ioc_writer. [Accessed: 20-Mar-2020].

[67] "CybOX," 2020. [Online]. Available: http://cyboxproject.github.io/ and https://github.com/CybOXProject. [Accessed: 20-Mar-2020].

[68] "Pyhton CybOX," 2020. [Online]. Available: `https://github.com/CybOXProject/python-cybox`. [Accessed: 20-Mar-2020].

[69] "MAEC," 2020. [Online]. Available: `http://maecproject.github.io/` and `https://github.com/MAECProject`. [Accessed: 20-Mar-2020].

[70] "MMDEF," 2011. [Online]. Available: `http://standards.ieee.org/develop/indconn/icsg/mmdef.html`. [Accessed: 20-Mar-2020].

[71] "CVE," 2020. [Online]. Available: `https://cve.mitre.org`. [Accessed: 20-Mar-2020].

[72] "CWE," 2020. [Online]. Available: `http://cwe.mitre.org`. [Accessed: 20-Mar-2020].

[73] "CPE," 2020. [Online]. Available: `https://nvd.nist.gov/products/cpe`. [Accessed: 20-Mar-2020].

[74] "CVSS," 2020. [Online]. Available: `http://www.first.org/cvss`. [Accessed: 20-Mar-2020].

[75] "CWSS," 2014. [Online]. Available: `http://cwe.mitre.org/cwss`. [Accessed: 20-Mar-2020].

[76] "CAPEC," 2020. [Online]. Available: `https://capec.mitre.org`. [Accessed: 20-Mar-2020].

[77] "STIX 1," 2018. [Online]. Available: `http://stixproject.github.io/`. [Accessed: 18-Apr-2020].

[78] "STIX 2," 2020. [Online]. Available: `https://oasis-open.github.io/cti-documentation/stix/intro`. [Accessed: 18-Apr-2020].

[79] Burger, E. W., Goodman, M. D., Kampanakis, P., and Zhu, K. A., "Taxonomy model for cyber threat intelligence information exchange technologies," in *Proceedings of the 2014 ACM Workshop on Information Sharing & Collaborative Security*, pp. 51–60, ACM, 2014.

[80] Sauerwein, C., Sillaber, C., Mussmann, A., and Breu, R., "Threat intelligence sharing platforms: An exploratory study of software vendors and research perspectives," *13th International Conference on Wirtschaftsinformatik*, 2017.

[81] Shackleford, D., "Who's using cyberthreat intelligence and how," *SANS Institute*, 2015.

[82] "STIX 1 Tools," 2018. [Online]. Available: `https://github.com/STIXProject`. [Accessed: 18-Apr-2020].

[83] "Malware Information Sharing Platform (MISP) on Github," 2020. [Online]. Available: `http://github.com/MISP/MISP/`. [Accessed: 27-Jun-2020].

[84] Tounsi, W. and Rais, H., "A survey on technical threat intelligence in the age of sophisticated cyber attacks," *Computers & Security*, vol. 72, pp. 212 – 233, 2018.

[85] "OpenIOC-to-STIX," 2018. [Online]. Available: `https://github.com/STIXProject/openioc-to-stix`. [Accessed: 20-Mar-2020].

[86] "Comparing STIX 1.x/CybOX 2.x with STIX 2," 2020. [Online]. Available: `https://oasis-open.github.io/cti-documentation/stix/compare#one-standard`. [Accessed: 18-Apr-2020].

[87] Danyliw, R., "The incident object description exchange format version 2," tech. rep., Internet Engineering Task Force (IETF), 2016.

[88] Takahashi, T., Landfield, K., and Kadobayashi, Y., "An incident object description exchange format (iodef) extension for structured cybersecurity information," tech. rep., Internet Engineering Task Force (IETF), 2014.

[89] "IETF Managed Incident Lightweight Exchange (MILE) Working Group," 2020. [Online]. Available: `https://datatracker.ietf.org/wg/mile/documents/`. [Accessed: 11-May-2020].

[90] "Malware Information Sharing Platform (MISP)," 2020 (last released June 26, 2020). [Online]. Available: `http://www.misp-project.org/`. [Accessed: 27-Jun-2020].

[91] "TLP," 2020. [Online]. Available: `https://www.first.org/tlp/`. [Accessed: 11-May-2020].

[92] "XCCDF," 2012. [Online]. Available: `http://scap.nist.gov/specifications/xccdf/`. [Accessed: 11-May-2020].

[93] Scarfone, K. and Mell, P., "The common configuration scoring system (ccss): Metrics for software security configuration vulnerabilities," *NIST interagency report*, vol. 7502, 2010.

[94] "SWID Tags," 2020. [Online]. Available: `http://tagvault.org/swid-tags/`. [Accessed: 11-May-2020].

[95] "The Trusted Automated eXchange of Indicator Information (TAXII) 2.1," 2020. [Online]. Available: `https://oasis-open.github.io/cti-documentation/taxii/intro`. [Accessed: 11-May-2020].

[96] "Extensible Messaging and Presence Protocol (XMPP)," 2020. [Online]. Available: `https://xmpp.org/`. [Accessed: 11-May-2020].

[97] Appala, S., Cam-Winget, N., McGrew, D., and Verma, J., "An actionable threat intelligence system using a publish-subscribe communications model," in *Proceedings of the 2Nd ACM Workshop on Information Sharing and Collaborative Security*, WISCS '15, (New York, NY, USA), pp. 61–70, ACM, 2015.

[98] "Extensible Markup Language (XML)," 2020. [Online]. Available: `https://www.xml.com/`. [Accessed: 11-May-2020].

[99] "JavaScript Object Notation (JSON)," 2020. [Online]. Available: https://www.json.org/. [Accessed: 11-May-2020].

[100] Skopik, F., Settanni, G., and Fiedler, R., "A problem shared is a problem halved: A survey on the dimensions of collective cyber defense through security information sharing," *Computers & Security*, vol. 60, pp. 154–176, 2016.

[101] "The MANTIS Cyber Threat Intelligence Management Framework," 2014. [Online]. Available: https://github.com/siemens/django-mantis. [Accessed: 16-Jun-2020].

[102] "MISP Default Feeds," 2020. [Online]. Available: http://www.misp-project.org/feeds/. [Accessed: 16-Jun-2020].

[103] "Collective Intelligence Framework (CIF)," 2020. [Online]. Available: https://github.com/csirtgadgets/bearded-avenger/. [Accessed: 16-Jun-2020].

[104] "CIF Default Feeds," 2020. [Online]. Available: https://github.com/csirtgadgets/bearded-avenger/tree/master/rules/default. [Accessed: 16-Jun-2020].

[105] "CIF modules," 2020. [Online]. Available: https://github.com/csirtgadgets/csirtg-smrt-py/tree/master/csirtg_smrt/parser, https://github.com/csirtgadgets/cif-sdk-stix-py. [Accessed: 16-Jul-2020].

[106] "Collaborative Research Into Threats (CRITs)," 2019. [Online]. Available: https://github.com/crits/crits/. [Accessed: 16-Jul-2020].

[107] "Combine indicator collecting tool," 2016. [Online]. Available: https://github.com/mlsecproject/combine, https://github.com/mlsecproject/combine/wiki/Threat-Intelligence-Feeds-Gathered-by-Combine. [Accessed: 16-Jul-2020].

[108] Nakamoto, S., "Bitcoin: A peer-to-peer electronic cash system," tech. rep., Manubot, 2019.

[109] Wüst, K. and Gervais, A., "Do you need a blockchain?," in *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pp. 45–54, IEEE, 2018.

[110] Cachin, C. and Vukolić, M., "Blockchain consensus protocols in the wild," *arXiv preprint arXiv:1707.01873*, 2017.

[111] Zheng, X., Zhu, Y., and Si, X., "A survey on challenges and progresses in blockchain technologies: A performance and security perspective," *Applied Sciences*, vol. 9, no. 22, p. 4731, 2019.

[112] Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., and others,, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the thirteenth EuroSys conference*, pp. 1–15, 2018.

[113] Hearn, M., "Corda: A distributed ledger," *Corda Technical White Paper*, vol. 2016, 2016.

[114] ConsenSys,, "Quorum Whitepaper," 2018. [Online]. Available: `https://github.com/ConsenSys/quorum/blob/master/docs/Quorum%20Whitepaper%20v0.2.pdf`. [Accessed: 17-Feb-2021].

[115] Wood, G. and others,, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1–32, 2014.

[116] Valenta, M. and Sandner, P., "Comparison of ethereum, hyperledger fabric and corda," *Frankfurt School Blockchain Center*, vol. 8, 2017.

[117] Polge, J., Robert, J., and Le Traon, Y., "Permissioned blockchain frameworks in the industry: A comparison," *ICT Express*, 2020.

[118] Dinh, T. T. A., Wang, J., Chen, G., Liu, R., Ooi, B. C., and Tan, K.-L., "Blockbench: A framework for analyzing private blockchains," in *Proceedings of the 2017 ACM International Conference on Management of Data*, pp. 1085–1100, 2017.

[119] Brandenburger, M., Cachin, C., Kapitza, R., and Sorniotti, A., "Blockchain and trusted computing: Problems, pitfalls, and a solution for hyperledger fabric," *arXiv preprint arXiv:1805.08541*, 2018.

[120] Gorenflo, C., Lee, S., Golab, L., and Keshav, S., "Fastfabric: Scaling hyperledger fabric to 20 000 transactions per second," *International Journal of Network Management*, vol. 30, no. 5, p. e2099, 2020.

[121] Nasir, Q., Qasse, I. A., Abu Talib, M., and Nassif, A. B., "Performance analysis of hyperledger fabric platforms," *Security and Communication Networks*, vol. 2018, 2018.

[122] Hyperledger,, "Hyperledger Fabric Documentation," 2020. [Online]. Available: `https://hyperledger-fabric.readthedocs.io/en/release-1.4/whatis.html`. [Accessed: 22-Jun-2020].

[123] Kampanakis, P., "Security automation and threat information-sharing options," *IEEE Security & Privacy*, vol. 12, no. 5, pp. 42–51, 2014.

[124] Steinberger, J., Sperotto, A., Golling, M., and Baier, H., "How to exchange security events? overview and evaluation of formats and protocols," in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, pp. 261–269, IEEE, 2015.

[125] Mavroeidis, V. and Bromander, S., "Cyber threat intelligence model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence," in *Intelligence and Security Informatics Conference (EISIC), 2017 European*, pp. 91–98, IEEE, 2017.

[126] Menges, F. and Pernul, G., "A comparative analysis of incident reporting formats," *Computers & Security*, vol. 73, pp. 87–101, 2018.

[127] Dandurand, L. and Serrano, O. S., "Towards improved cyber security information sharing," in *2013 5th International Conference on Cyber Conflict (CYCON 2013)*, pp. 1–16, IEEE, 2013.

[128] Sillaber, C., Sauerwein, C., Mussmann, A., and Breu, R., "Data quality challenges and future research directions in threat intelligence sharing practice," in *Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security*, pp. 65–70, ACM, 2016.

[129] Appala, S., Cam-Winget, N., McGrew, D., and Verma, J., "An actionable threat intelligence system using a publish-subscribe communications model," in *Proceedings of the 2nd ACM Workshop on Information Sharing and Collaborative Security*, pp. 61–70, ACM, 2015.

[130] de Fuentes, J. M., González-Manzano, L., Tapiador, J., and Peris-Lopez, P., "Pracis: Privacy-preserving and aggregatable cybersecurity information sharing," *Computers & Security*, vol. 69, pp. 127–141, 2017.

[131] Riesco, R., Larriva-Novo, X., and Villagra, V., "Cybersecurity threat intelligence knowledge exchange based on blockchain," *Telecommunication Systems*, pp. 1–30, 2019.

[132] Homan, D., Shiel, I., and Thorpe, C., "A new network model for cyber threat intelligence sharing using blockchain technology," in *2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, pp. 1–6, IEEE, 2019.

[133] Hajizadeh, M., Afraz, N., Ruffini, M., and Bauschert, T., "Collaborative cyber attack defense in sdn networks using blockchain technology," in *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, pp. 487–492, IEEE, 2020.

[134] Ponemon Institute, L., "Third Annual Study on Exchanging Cyber Threat Intelligence: There Has to Be a Better Way," 2018. [Online]. Available: `https://www.infoblox.com/wp-content/uploads/infoblox-white-paper-ponemon-infoblox-2018-final-report.pdf`. [Accessed: 17-Jun-2020].

[135] Bourgue, R., Budd, J., Homola, J., Wlasenko, M., and Kulawik, D., "Detect , SHARE , Protect Solutions for Improving Threat Data Exchange among CERTs," *European Network and Information Security Agency (ENISA)*, no. October, p. 51, 2013.

[136] Ring, T., "Threat intelligence: why people don't share," *Computer Fraud & Security*, vol. 2014, no. 3, pp. 5–9, 2014.

[137] Johnson, C., Badger, M., Waltermire, D., Snyder, J., and Skorupka, C., "Guide to cyber threat information sharing," tech. rep., National Institute of Standards and Technology, 2016.

[138] Voigt, P. and Von dem Bussche, A., “The eu general data protection regulation (gdpr),” *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 2017.

[139] ENISA,, *ENISA Threat Landscape Report 2016: 15 Top Cyber-Threats And Trends*. ENISA, 2017.

[140] Hyperledger,, “Hyperledger Fabric Prerequisites,” 2020. [Online]. Available: `https://hyperledger-fabric.readthedocs.io/en/release-1.4/prereqs.html`. [Accessed: 12-Jun-2020].

[141] “CIRCL OSINT Feeds,” 2020. [Online]. Available: `https://www.circl.lu/doc/misp/feed-osint`. [Accessed: 16-Jun-2020].

[142] Thakkar, P., Nathan, S., and Viswanathan, B., “Performance benchmarking and optimizing hyperledger fabric blockchain platform,” in *2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pp. 264–276, IEEE, 2018.

[143] Chung, G., Desrosiers, L., Gupta, M., Sutton, A., Venkatadri, K., Wong, O., and Zugic, G., “Performance tuning and scaling enterprise blockchain applications,” *arXiv preprint arXiv:1912.11456*, 2019.

[144] Heyman, J., Byström, C., Hamrén, J., and Heyman, H., “Locust,” 2020. [Online]. Available: `https://docs.locust.io/en/stable/what-is-locust.html`. [Accessed: 15-Jun-2020].

[145] Hyperledger,, “Hyperledger Fabric Performance Reports,” 2021. [Online]. Available: `https://hyperledger.github.io/caliper-benchmarks/`. [Accessed: 12-Apr-2021].

# APPENDIX A

# CHAINCODE

```
const { Contract } = require('fabric-contract-api');

class CyberEvent extends Contract {

  async instantiate(ctx) {
    console.info('== Initialize Ledger ==');
  }

  async createCyberEvent(ctx, id, eventObj) {
    console.info('== START : Create Cyber Event ==');

    const docType = 'cyberEvent';
    const key = ctx.stub.createCompositeKey(docType, [id]);

    const cyberEvent = Object.assign({}, JSON.parse(eventObj));

    cyberEvent['id'] = id;
    cyberEvent['docType'] = docType;
    if(cyberEvent.Tag[0].name=='tlp:white'){
      console.info('== Tag is suitable == ');
      await ctx.stub.putState(key,
                  Buffer.from(JSON.stringify(cyberEvent)));
      console.info('== END : Create Cyber Event ==');
    }else{
      console.info('== END: Tag is not suitable ==');
    }
  }

  async queryCyberEvent(ctx, id) {
    console.info('== START : Query Cyber Event ==');
    const docType = 'cyberEvent';
    const key = ctx.stub.createCompositeKey(docType, [id]);
    try{
      const cyberEventAsBytes = await ctx.stub.getState(key);
      console.info('== END : Query Cyber Event ==');
      return cyberEventAsBytes.toString();
    }
    catch(e){
      return null;
    }
  }

  async queryCyberEventsRange(ctx, startId, endId) {
    console.info('== START : Query Cyber Events Range ==');
    const docType = 'cyberEvent';
    const startKey = ctx.stub.createCompositeKey(docType, [startId]);
    const endKey = ctx.stub.createCompositeKey(docType, [endId]);
    const iterator = await ctx.stub.getStateByRange(startKey, endKey);
    const allResults = [];
    while (true) {
      const res = await iterator.next();
      if (res.value && res.value.value.toString()) {
        const Key = res.value.key;
        let Record;
        try {
          Record = JSON.parse(res.value.value.toString('utf8'));
        } catch (err) {
          console.log(err);
          Record = res.value.value.toString('utf8');
        }
          allResults.push({ Key, Record });
        }
        if (res.done) {
          await iterator.close();
          console.info('== END : Query Cyber Events Range ==');
          return allResults;
        }
    }
  }

}

module.exports = CyberEvent;
```

# APPENDIX B

## SAMPLE EVENT

```
{
  "Event": {
    "info": "AutoIT-compiled Negasteal/Agent Tesla",
    "publish_timestamp": "1572554081",
    "timestamp": "1572554076",
    "analysis": "2",
    "Attribute": [
      {
        "comment": "Troj.Win32.TRX.XXPE50FFF032",
        "category": "Payload delivery",
        "uuid": "5da1e919-6eb0-4063-996d-9c7ec0a8ab16",
        "timestamp": "1572554060",
        "to_ids": true,
        "value": "bc077b31c61d61d5d077b68b7f0b110efe85d138",
        "object_relation": null,
        "type": "sha1"
      },
      {
        "comment": "Troj.Win32.TRX.XXPE50FFF032",
        "category": "Payload delivery",
        "uuid": "5da1e919-7d04-4aed-9587-9c7ec0a8ab16",
        "timestamp": "1572554076",
        "to_ids": true,
        "value": "224f6e0c21145534ec2bab670bcb1b690c08a26d",
        "object_relation": null,
        "type": "sha1"
      },
      {
        "comment": "",
        "category": "External analysis",
        "uuid": "5da1e940-a8d8-4859-9f72-a060c0a8ab16",
        "timestamp": "1570892096",
        "to_ids": false,
        "value": "https://blog.trendmicro.com/trendl...",
        "object_relation": null,
        "type": "link"
      }
    ],
    "Tag": [
        {
          "colour": "#00d622",
          "exportable": true,
          "name": "tlp:white"
        }
      ],
      "published": true,
      "date": "2019-10-25",
      "Orgc": {
        "uuid": "56c42374-fdb8-4544-a218-41ffc0a8ab16",
        "name": "CUDESO"
      },
      "threat_level_id": "2",
      "uuid": "5da1e8e9-5614-4e68-b67a-a05dc0a8ab16"
    }
  }
```

**TEZ İZİN FORMU /** THESIS PERMISSION FORM

**ENSTİTÜ /** INSTITUTE

| | |
|---|---|
| **Fen Bilimleri Enstitüsü** / Graduate School of Natural and Applied Sciences | ☐ |
| **Sosyal Bilimler Enstitüsü** / Graduate School of Social Sciences | ☐ |
| **Uygulamalı Matematik Enstitüsü** / Graduate School of Applied Mathematics | ☐ |
| **Enformatik Enstitüsü** / Graduate School of Informatics | X |
| **Deniz Bilimleri Enstitüsü** / Graduate School of Marine Sciences | ☐ |

**YAZARIN /** AUTHOR

| | |
|---|---|
| **Soyadı** / Surname | : Özdemir |
| **Adı** / Name | : Ahmet |
| **Bölümü** / Department | : Siber Güvenlik |

**TEZİN ADI /** TITLE OF THE THESIS (**İngilizce** / English) :

Cyber Threat Intelligence Sharing Technologies and Threat Sharing Model using Blockchain

**TEZİN TÜRÜ /** DEGREE: **Yüksek Lisans** / Master X **Doktora** / PhD ☐

1. **Tezin tamamı dünya çapında erişime açılacaktır. /** Release the entire work immediately for access worldwide. X
2. **Tez iki yıl süreyle erişime kapalı olacaktır.** / Secure the entire work for patent and/or proprietary purposes for a period of **two year**. * ☐
3. **Tez altı ay süreyle erişime kapalı olacaktır.** / Secure the entire work for period of **six months**. * ☐

* *Enstitü Yönetim Kurulu Kararının basılı kopyası tezle birlikte kütüphaneye teslim edilecektir.*
*A copy of the Decision of the Institute Administrative Committee will be delivered to the library together with the printed thesis.*

**Yazarın imzası** / Signature ............................ **Tarih** / Date 7.5.2021