Study and Design of Blockchain Based Decentralized Trust Management System for Secure Transactions

Varun Deshpande

ÉCOLE DOCTORALE UNIVERSITÉ PARIS-EST

Mathématiques et STIC

THESIS

to obtain the title of

Doctor of Philosophy

Specialization: Computer Science

by

Varun Deshpande

Study and Design of Blockchain Based Decentralized Trust Management System for Secure Transactions

---

defended on December 17, 2020

Jury

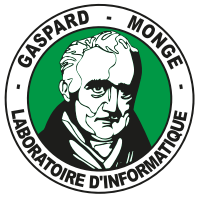
Advisors: **Laurent George** - Professor, Université Gustave Eiffel
Hakim Badis - Asso. Professor, Université Gustave Eiffel

Reviewers: **Maria Potop-Butucaru** - Professor, Sorbonne Université
Hakima Chaouchi - Professor, Télécom SudParis

Examiners: **Dorian Grosso** - Head of R&D, METRON SAS, Paris
Ye-Qiong Song - Professor, Université de Lorraine

Ph.D. prepared at
ESIEE Paris
2 boulevard Blaise Pascal,
Cité Descartes BP 99,
93162 Noisy-le-Grand Cedex, France

Ph.D. in collaboration with
METRON
30 rue de Gramont,
75002 Paris, France



Sous la co-tutelle de :
CNRS
ÉCOLE DES PONTS PARISTECH
UNIVERSITÉ GUSTAVE EIFFEL

Ph.D. in collaboration with
LIGM
Laboratoire d'Informatique Gaspard-Monge
Cité Descartes, Batiment Copernic
5 boulevard Descartes,
77454 Marne-la-vallée Cedex 2, France

न चोराहार्यम् न च राजहार्यम्,
न भ्रातृभाज्यं न च भारकारि ।

व्यये कृते वर्धत एव नित्यं,
विद्याधनं सर्वधनप्रधानम् ।

It cannot be stolen by thieves,
nor can it be taken away by the kings.

It cannot be divided among brothers,
nor it has a weight.

If spent regularly,
it will always keep growing.

The wealth of knowledge is
the most superior wealth of all.

Acknowledgments

First and foremost, I am greatly indebted to my advisor, Prof. Laurent George, who throughout my journey of 5 years at ESIEE Paris, has provided me with everything I could have ever asked for. From accepting me as his intern during my Engineering to guiding me through my Masters and Ph.D. degree, he has always been kind and supportive to me. It was due to his constant motivation that I have successfully finished my Ph.D. at the age of 26. His research rationale with clear focus was truly a guiding light during my Ph.D. I am very grateful to him for the thought-provoking discussions, sharing his broad knowledge and experience in every aspect of IoT and blockchain.

It is a privilege for me to have worked with Prof. Hakim Badis, my co-advisor, who has constantly shared his valuable knowledge and helped me solve some of the most complex problems in Blockchain. He is extremely knowledgeable in the field of graph theory and I am truly grateful to him for his tremendous help in my research and countless support in my Ph.D. work.

I am also grateful to University Paris-Est and Metron for co-financing my Ph.D. research. Special thanks goes to Mr David Tagliabue (President, METRON) and Dorian Grosso (Head of R&D, METRON) for their priceless support throughout my thesis. Mrs Sylvie Cache (Secretary, Université Gustave Eiffel - LIGM) should also be thanked for her compassionate support for all the administrative formalities over 3 years.

A huge thanks to all my friends. It has been a great privilege to be able to associate with them on many research projects during my Ph.D., and have countless fun moments in life, which I will always cherish. Special thanks to all my close friends for always supporting me. You all have a special place in my heart and I hope our friendship lasts forever.

Most importantly, I would like to thank my beloved parents from the bottom of my heart for their unconditional love, support and guidance throughout my life. This thesis would not have been possible without your support all along the way. Lastly, I would like to thank the almighty God for giving me the strength, knowledge, ability and opportunity to undertake this research study and to persevere and complete it satisfactorily.

Abstract

Study and Design of Blockchain Based Decentralized Trust Management System for Secure Transactions

In the context of modern connected world, the concept of atomic data transfer/transaction has been completely redefined. Traditional distributed databases solve the issue of data safety through classical Atomicity, Consistency, Isolation and Durability (ACID) properties. However, the complex issue of transaction security remains difficult to address. Introducing blockchain (BC) as distributed database partially solves the problem but another issue arises i.e., BC's modeling and evaluation. For e.g., what parameter values are ideal, is the selected blockchain framework compatible, etc.

In this thesis, we solve it through dimensioning of BC using graph theory. With binomial distribution and preferential attachment models, we model the underlying BC P2P network to reduce topology control overhead while ensuring high flexibility, fast reconfigurability, connectivity, small diameter and clustering. Next, to reduce the no. of connections per peer, we establish ideal bounds on outbound and inbound connections that still guarantee P2P network feasibility and connectivity using r-out digraphs. For an already developed BC framework, we evaluate its applicability through topology mapping. We demonstrate the efficiency of our approach using our BTCmap framework applied to Bitcoin and present its real captured snapshot.

However, using BC alone cannot holistically secure transaction as it only guarantees data immutability whereas in most scenarios, the data has also to be secured at point of generation and usage. Further, BC has high overhead and cannot penetrate to lower levels in a system. To mitigate this, we propose the use of Secure Element (SE) to establish "root of trust", following the "secure by design" paradigm. Using these two technologies as the base of our proposed decentralized system, we apply it to three disparate fields. In Smart Grids, we address the problem of designing of distributed marketplace using the concepts of blockchain, SE, applied smart contracts, escrow accounts. We also address the issue of large data storage on blockchain for DR and centralization in DR allotment by designing a decentralized autonomous bidding system. We also propose a fair and efficient DR allotment algorithm whose execution time is less than 1 minute for more than 20k participants.

Next, we apply our BC-SE based **Sa**feguarding **F**ram**e**work (SaFe) to smart vehicles and address the problem of insecure Execution and Storage Environment within the smart vehicle. With SaFe, we show, how non-repudiable responsibility can be enforced. This ensures that when regulators audit the data, its veracity is undeniable. The issue of secure firmware update and key management is also solved while improving performance. In IoT, we apply our SE BC Stratagem (SEBS) to solve the pressing issue of holistic data security in resource constrained devices i.e., securing data at all 3 points viz., generation, storage, and usage while maintaining very low overhead and improving performance. We also address the niche issue of verification of blockchain data where a remote device which receives data from blockchain through an intermediary, does not have resources and online connectivity to verify it. By proposing SEOVA's double signature algorithm using the technology of SE, we successfully solve this issue without compromising on security and privacy.

In the last leg, we solve the classical problem of sensor monitoring where data is transmitted using "limited transmit" method which results in aperiodicity in transmit patterns for various onsite sensors. The system on the other end must distinguish between two scenarios i.e. 1) data is not transmitted because transmission conditions are unmet, and 2) data is not transmitted because of an error on remote site. Given the same outcome for both cases, our SE-based PulSec framework, cleverly solves this problem while maintaining extremely low overhead (bandwidth=7.73 B/s, memory=464 B, time=500 ms) and zero false-positive cases.

Keywords: Blockchain, Secure Element, Internet of Things, Smart Vehicle, Smart Grid, Energy Efficiency, Demand Response, Non-repudiable Responsibility

Résumé

Étude et conception d'un système de gestion décentralisée de la confiance basé sur blockchain pour des transactions sécurisées

Les bases de données distribuées traditionnelles résolvent le problème de la sureté des données grâce aux propriétés classiques d'atomicité, de cohérence, d'isolation et de durabilité (ACID) des transactions en charge de la mise à jour de ces données. Cependant, la question complexe de la sécurité des transactions reste difficile à résoudre. L'introduction de la chaîne de blocs (BlockChain : BC) en tant que base de données distribuée résout partiellement le problème, mais une autre question se pose, à savoir la modélisation et l'évaluation de la performance de la BC.

Dans cette thèse, nous étudions le problème de dimensionnement de la BC en utilisant la théorie des graphes. À l'aide d'un modèle de distribution binomiale pour la sélection des nœuds voisins, nous modélisons le réseau P2P sous-jacent d'une BC pour le contrôle de la topologie du réseau P2P. Ensuite nous caractérisons le nombre de connexions sortantes et entrantes qui garantissent la connectivité du réseau P2P à l'aide de digraphes r-out. Nous démontrons l'efficacité de notre approche en utilisant notre outil BTCmap appliqué à un réseau existant.

L'utilisation de la BC seule ne peut pas sécuriser les transactions de manière globale car elle ne garantit que l'immuabilité des données. Cependant, les données doivent également être sécurisées à leur source. De plus, la BC a un cout élevé en termes de puissance de calcul et ne peut pas être utilisée au niveau des nœuds sources des données. Pour traiter ce problème, nous proposons d'utiliser le concept d'élément de sécurité (SE) pour établir une "chaine de confiance" selon le paradigme de la "sécurité par conception".

Nous appliquons notre approche SE+BC à trois différents contextes :

-Dans le cadre des réseaux intelligents, nous abordons le problème de conception d'un marché distribué pour l'énergie en utilisant les concepts de contrats intelligents et de comptes séquestre. Nous abordons également le problème du stockage de données sur la chaîne de blocs pour la proposition

d'un service d'effacement énergétique décentralisé.

-Ensuite, nous nous nous nous attaquons au problème de l'environnement d'exécution et de stockage non sécurisé dans un véhicule intelligent. Avec la solution SaFe, nous montrons comment la responsabilité non-réfutable peut être mise en œuvre. Cela garantit que lorsque les régulateurs vérifient les données, leur véracité est indéniable. La question de la mise à jour sécurisée des microprogrammes et de la gestion des clés de chiffrement est résolue par la solution SaFe.

-Dans le contexte de l'Internet des Objets, nous appliquons notre stratégie SE-BC dans la solution SEBS, pour résoudre le problème de la sécurité globale des données dans les dispositifs à ressources limitées. Nous sécurisons les données en trois points, à savoir le point de génération, le stockage et l'utilisation avec un cout faible. Nous abordons également la question de la vérification des données de la BC lorsqu'un appareil distant qui reçoit des données de la BC par un intermédiaire ne dispose pas des ressources et de la connectivité en ligne nécessaires pour les vérifier. En proposant l'algorithme de double signature de SEOVA utilisant la technologie de SE, nous proposons une solution à ce problème sans compromettre la sécurité et la confidentialité.

Pour finir, nous résolvons le problème classique de la surveillance de capteurs où les données sont transmises en utilisant une méthode de "transmission limitée" qui conduit à une apériodicité dans les transmissions de divers capteurs sur site. Le système à l'autre bout de la chaine qui récupère les données doit distinguer deux scénarios : 1) les données ne sont pas transmises parce que les conditions de transmission ne sont pas remplies, c.a.d. pas de changement des données, et 2) les données ne sont pas transmises en raison d'une erreur sur un site distant. Avec le même résultat pour les deux cas, notre solution PulSec, basé sur les SE, résout ce problème avec un cout très faible.

Mots-clés : Blockchain, Elément de sécurité, Efficacité énergétique, Effacement énergétique, Internet des Objets, Véhicules intelligents

Contents

List of Acronyms

List of Algorithms

List of Figures

List of Tables

Chapter 1

Introduction

Contents

1.1 Introduction

In this modern era, the barrier between the physical world and the digital world is diminishing. As more and more things are being digitized, many traditional concepts from the physical world are being reinterpreted. This reinterpretation is redefining these traditional concepts, often making them more holistic. Concepts like signatures, currencies and fingerprinting are few of them.

Another concept, i.e., *transaction*, has been completely redefined. Merely 5 decades ago, a *transaction* simply meant currency transfer from one party to another. However, the present day *transaction* is no more related to finance domain only. The concept of *transaction* is actively used to signify atomic data transfer between parties. With digitization, this atomic data transfer can signify anything. For e.g., it can mean a financial data, sensor data or output of a code. Consequently, a *transaction* can be anything from a financial transaction to data transaction in a database or a sensor-data transaction in an Internet of Things (IoT) network or even a generic transaction in a blockchain.

The extent of application of the concept of transaction as an atomic data transfer is vast and diverse. As the concept of *transaction* has become more

holistic, to guarantee its fool proof execution, many additional properties like security, immutability, authentication, consensus oriented decisions, decentralization, Trusted Execution Environment (TEE), Trusted Storage Environment (TSE), monitoring, etc. are to be satisfied apart from the classical Atomicity, Consistency, Isolation, and Durability (ACID) properties.

Classical distributed databases satisfy ACID properties as well as guarantee distributed availability along with decentralization. These all can be grouped under the category of data safety. However, full *transaction* (data) security is still far from being solved. Given the complexity the security issue adds to any base system, no single technology fits for all scenario implementations.

1.2 Background

In the context of a distributed database, the atomic data transfers should satisfy the classical ACID properties. However, there is one additional aspect which has risen to prominence i.e., security. With the increase in the no. of connected devices, the need for secure atomic data transfer (*transaction*) is indispensable. At present, the biggest chunk of data *transactions* is coming from IoT networks.

Distributed databases, while solving the problem of safety of these atomic transfers/*transactions*, overlook their security aspect. The security aspect, even if simple to understand, is difficult to implement in a reliable way. There are various problems w.r.t. security such as modification of *transactions*, non-forwarding of *transactions*, forwarding fake *transactions*, etc. Hence, the addition of security paradigm increases the complexity of the system.

Going further, using blockchain technology as distributed database solves many of the security problems as it offers many important properties such as:

- Consensus oriented decisions.
- Guarantee of total order of *transactions*.
- Reliable multi-cast.
- Irreversible and immutable *transactions*.

However, there is an additional need for traceability of data and its origin which is not natively supported by blockchain or distributed database. This support can be easily added with the help of the technologies like Secure Element (SE). Moreover, as shown later in this thesis, SE can prevent data modification at source itself thereby preventing information modification.

This is important as any modification can restrict consensus which is integral for blockchain's functioning.

Much literature is available on securing various systems using blockchain or SE. However, both technologies have many impediments when used individually to secure diverse types of systems. Having realized this during the research course of this thesis, the focus of securing systems has been completely redefined. Instead of depending on any one technology for guaranteeing holistic security for *transactions*, this thesis focuses on a system which is agile and agnostic for application to many emerging fields and at the same time, avoid impediments of previously proposed research literature.

The proposed system in this thesis is blockchain-based with integral addition of SE technology at appropriate levels to address impediments. Thus, the resulting system is not only decentralized but also capable of managing trust w.r.t many application domains.

For the sake of easy understanding and coherency, the next three sections introduce the transaction classification, the concept of SE and blockchain.

1.3 Transactions' Classification

For easy understanding and organizational purpose in this thesis, *transactions* are divided into two types i.e.,transactions without history - *data transactions* and transactions with history - *financial transactions*. Transactions from database(s), sensor(s), IoT network(s) can be independent and may or may not be related to past transactions are essentially termed as *data transactions*. For e.g., power consumption data sent by a power sensor is independent and does not require previous values for ascertaining its validity.

On the other hand, transactions involving token/asset transfers which need verification of previous transactions trail are classified as *financial transactions*. For e.g. Person A transfers 100 tokens to Person B. This transfer can only be verified by previous transaction trail of Person A to ascertain if he/she has enough tokens to realize this transfer.

As the properties of these two types of *transactions* are different, designing a system for securing both of them becomes trickier. For e.g., designing a system to secure *financial transactions* may not be a difficult task as it involves tracing a transaction trail. However, the dimension of difficulty rises manifold once we decide to design it for a resource constraint system (like remote IoT devices). Similarly, for *data transactions*, ascertaining its validity is an arduous task as there is no transaction trail for the same. Moreover, if the source is a resource constraint system, like a remote IoT device, ascertaining its authenticity becomes as important as its validity.

1.4 Introduction to Blockchain

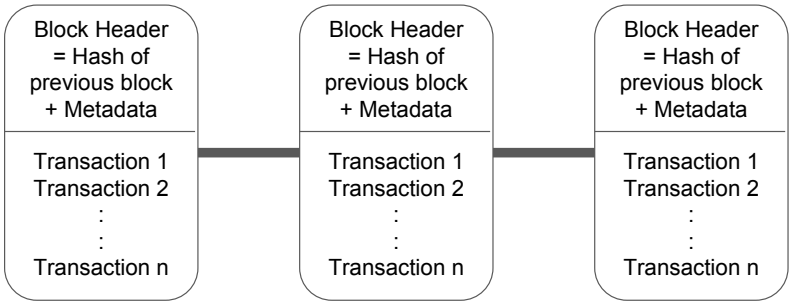


Figure 1.1: A basic blockchain structure

Blockchain, in simple terms, as the name suggests is a series of blocks (containing data) in which blocks are linked to each other via cryptographic functions forming a strong chain. This helps it to record transactions through a secure and verifiable process without any intermediary [Deshpande *et al.* 2018]. In most cases, it is only the hash of the previous block which is included in the current block while its formation. Blockchain as a concept is nearly 3 decades old. The first implementation of blockchain concept was presented in 1990 by Haber et al in their work titled "How to timestamp a document" [Haber & Stornetta 1990]. However, blockchain was brought to prominence in 2008 by Bitcoin [Nakamoto *et al.* 2008], the first cryptocurrency implementation based on blockchain. Since then, the economics of blockchain-cryptocurrency have scaled up with the estimated combined market cap of $175 Billion [CoinMarketCap 2018] (April 2019).

In its simplest form, a block in the blockchain contains data segregated in multiple small entities called as transactions, a hash of the previous block. The structure is explained in Figure 1.1. Blockchain is based on Distributed Ledger Technology and achieves immutability by securely storing ledger copies on all the participating nodes. The coherency is achieved through different consensus algorithms (consensus based on Proof of Work (PoW), Proof of Stake (PoS), etc.). The participating nodes form the Peer to Peer (P2P) network and a communication protocol is implemented among them to carry out information dissemination.

Blockchain classification is explained in [Deshpande *et al.* 2019c]. Blockchain can be divided into many types depending on various parameters like permissions, application domain, protocols, etc. Based on permissions, Blockchain can be classified into two main categories i.e. permissioned and non-permissioned. In a permissioned blockchain, prerequisite permissions are needed to connect/read/write on the blockchain while with non-permissioned,

no explicit permissions are needed. The classification flow in Figure 1.2 explains in detail about various nomenclature surrounding this classification axis.

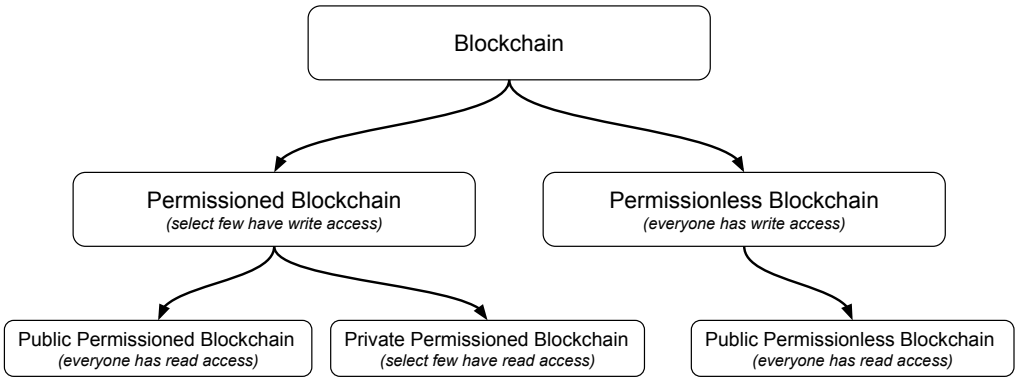


Figure 1.2: Blockchain classification based on permissions

The two major categories of blockchain based on application domains are 1) Transaction Optimized Blockchains and 2) Code Optimized Blockchains. Transaction Optimized Blockchains are mainly used for the transfer of assets and to some extent for data storage while code optimized blockchains are mainly used for the execution of code on the blockchain. The overhead of code optimized blockchain is generally higher than transaction optimized blockchain because of the fact that there are additional virtual machines running on each participating node for the execution of code and also there are additional messages for sharing the current state of variables.

1.4.1 Blockchain P2P Network Properties

Following are the important blockchain P2P network properties which have been considered in this thesis:

1.4.1.1 Topology Control

The topology control mainly includes two operations: topology construction and topology maintenance. The topology construction is related to peers discovery and neighbors selection while topology maintenance is related to re-constructing the topology when nodes join/leave the network or when changes are proposed in existing blockchain frameworks to build the global blockchain P2P network. When a peer wants to join a blockchain P2P network, it starts by discovering other participating peers that are already connected. The IP

addresses of the peers found are stored locally in a list. Then, the peer randomly selects r neighbors from the list to establish and maintain its outbound connections. When an outbound connection fails, another neighbor is selected and a new outbound connection is initiated. A distributed topology control suffers from two difficulties: overhead and lack of flexibility.

1.4.1.1.1 Overhead

The distributed topology control operations involve extra overhead in terms of communication traffic which results in significant consumption of network bandwidth, energy, memory and CPU usage. Thus, resource-limited devices such as IoT devices, smartphones, etc., do not support blockchain capabilities or, at worst, run a blockchain with limited functionalities. For example, Long Range Wide Area Network (LoRaWAN) [de Carvalho Silva *et al.* 2017] or SigFox [Lavric *et al.* 2019] end-devices have a small battery, a low data rate (limited number of exchanged messages per day and short payload size) and a low computing power and storage. They cannot perform by themselves the topology control operations. To deal with this issue, we propose a centralized topology control where a secure overlay layer composed of distributed and synchronized servers handles and manages the topology control operations instead of the participating peers. The vertical plane (peers-servers) is dedicated to the topology control traffic while the horizontal plane (inter-peer) is dedicated to data traffic control.

1.4.1.1.2 Flexibility

In a blockchain ecosystem, the ability to adapt and include changes on-demand is defined as flexibility. In traditional blockchain P2P network, any change in the topology control functions/configuration such as the peer discovery process, the number of outbound/inbound connection or the neighbor selection strategy, is handled by each participating peer. This operation requires more time, especially for large-scale public blockchains, to allow all peers updating the installed blockchain software. When some peers (including key peers) take time to update the configurations, the global blockchain P2P network properties can be impacted and the performance can be degraded.

The idea of re-configurable networks has received considerable attention in recent years due to the emergence of the Software-Defined Networking (SDN) paradigm where a separation is done between the data plane and the control plane. The control plane is placed in a centralized controller. Thus, any changes in the topology control functions/configuration will only be implemented at the controller and the resulting configuration is injected in the

peers. In our work, as explained later in Section 2.2.3, we propose to use the SDN paradigm where the overlay layer of distributed and synchronized servers can be viewed as like a centralized controller in SDN.

1.4.1.2 Connectivity

For any underlying P2P network, *connectivity* is necessary for a number of reasons. Proper connectivity helps not only in faster information dissemination but also to maintain a single coherent blockchain by reducing the probability of fork formation that can be the source of attacks as explained in the following example. Figure 1.3 shows a simple view of a blockchain after the P2P network becomes disconnected and forms two independent sub-networks. We can see that each independent sub-network adds new subsequent blocks and considers the new blockchain fork as the main blockchain. In this case, a malicious peer can efficiently perform a double-spending attack on each sub-network. Moreover, it is possible that the isolated sub-network lacks even one full node, thereby making the blockchain fork invalid within that sub-network as full transaction history from genesis block is absent. Further, when the number of isolated sub-networks within a P2P network increases, full nodes are unequally split across them. This increases the success rate of Distributed Denial-of-Service Distributed Denial-of-Service (DDoS) attack which works by flooding full nodes of a targeted isolated sub-network.

When the two isolated sub-networks are merged into one network, one blockchain appears with two forks, out of which, the longest chain is considered in the main chain. This effectively reverses the transactions in the other fork which may lead to financial repercussions for the businesses depending on blockchain.(e.g., real-world goods delivery, on-demand services, etc.) In our work, as our blockchain P2P network topology is a r-*out* digraph, we derive the smallest value of r to guarantee the network connectivity and avoid such forks altogether.

1.4.1.3 Smaller Diameter

Smaller diameter of the P2P network helps in quicker information dissemination and thereby minimizes the number of transient forks in a given blockchain. Fewer forks increase overall trustability of the blockchain and reduce the mining time in the blockchain (and also to assure that the transaction is indeed included in the main chain). Smaller diameter can also ensure smaller block propagation time thereby enabling the participants either to add more blocks in the given period or increase the block size, both without increasing the number of transient forks. It also aids in quicker consensus time.

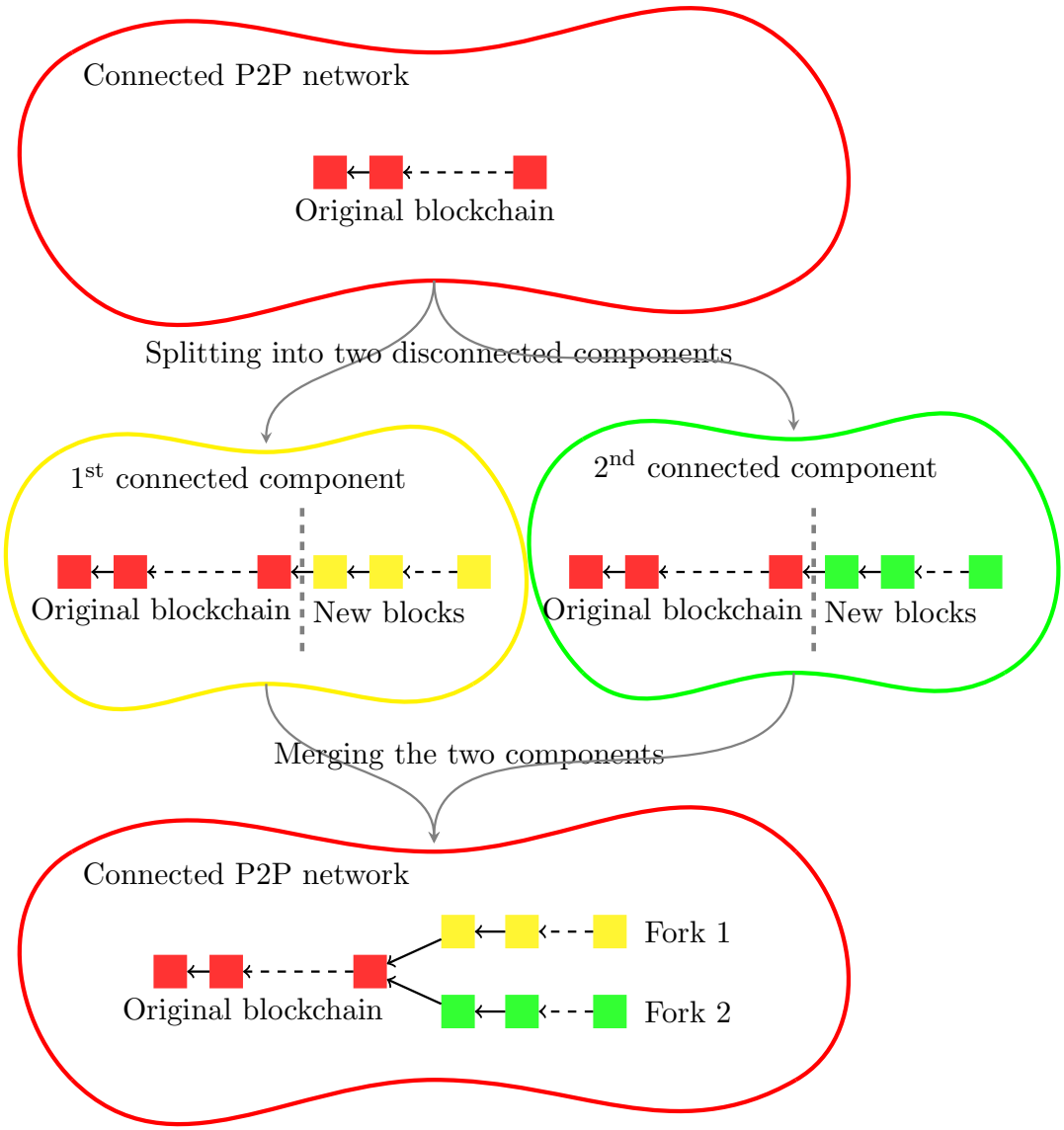


Figure 1.3: Example of a blockchain evolution when splitting and merging a connected P2P network.

1.4.1.4 Consensus Efficiency

The goal of blockchain consensus protocols is to maintain the exact same copy of the ledger on all honest peers, tolerating a bounded number of byzantine faults [Lamport *et al.* 1982]. To reach consensus, all consensus algorithm require at least a connected P2P network [Wang *et al.* 2019]. The existence of a solution to a consensus problem depends on the hypothesis we make on communication delays and on processor speeds:

- Synchronous: the worst case communication delays are bounded and the bounds are known. The processor speeds are bounded and known.

- Partially Synchronous: The worst case communication delays are bounded but the bounds are not known, or the bounds are known but after an unknown period of time, or the bounds are valid during a sufficient period of time for the consensus to be completed. Same for the processors speeds.

- Asynchronous: The communication delays are not bounded and can be infinite. The processor speeds are unknown.

An important result shows that the consensus problem cannot be solved in the asynchronous case (assuming no global time reference is available, i.e. no clock synchronization protocol) even in the case of a single node crash [Fischer *et al.* 1985]. In our work, we therefore consider the partially synchronous case. We also suppose message authentication. In the case of partially synchronous communications with authentication, several consensus protocols have been proposed [Dwork *et al.* 1988]. These protocols vary with different blockchain networks [Wang *et al.* 2019] and can be classified as voting-based consensus and proof-based consensus algorithms.

The voting-based consensus algorithms are based on a cooperative approach between a subset of *identity-verified* participating peers and are mainly (but not only) applied for permissioned or private blockchain networks. Examples of such algorithms are Practical Byzantine Fault Tolerance (PBFT) (tolerates at most f faulty/malicious nodes for $3f$+1 total nodes) [Castro & Liskov 1999], Crash Fault Tolerance (CFT) [Baldoni *et al.* 2000], etc. The proof-based consensus algorithms are based on a competition approach between participating peers and are mainly (but not only) applied for non-permissioned blockchain networks. Examples of such algorithms are PoW [Nguyen & Kim 2018], PoS [Nguyen & Kim 2018], Proof of Elapsed Time (PoET) [Chen *et al.* 2017], Proof of Burn (PoB) [Karantias *et al.* 2020], etc.

Indeed, while voting-based consensus assume fully connected P2P network between voting peers [Wang *et al.* 2019], proof-based consensus algorithms assume (simply) connected P2P networks [Gervais *et al.* 2015, Pasqualetti *et al.* 2012]. In our work, in order to keep the consensus efficiency, we maintain the connectivity conditions according to the blockchain type. Further, despite our proposed architecture being two-layered, there are no additional security implications arising out of this. This is because the main blockchain functioning is still carried out in a single layer (lower-layer)

only thereby making no difference w.r.t. consensus application in blockchain's working.

1.4.1.5 Clusters

In graph theory, clustering refers to the task of grouping nodes into clusters, so that the edge density is higher within clusters and relatively lower between clusters. Generally, the clusters appear in large-scale public blockchains due to the presence of peers (called super-peers) having more resources to perform the consensus algorithm, relay data between clusters, etc. In our work, we propose to use a centralized network generator at the topology control layer to manage the blockchain P2P network clustering. Thus, the number of clusters can be dynamically adjusted according to the desired performance.

1.5 Introduction to Secure Element

A Secure Element (SE) [GlobalPlatform, Inc. 2018] is a programmable micro-controller which is tamper-resistant and provides a TEE [Jurgensen & Guthery 2002] and TSE. SEs are generally small in size (25 mm^2) and designed to support security operations like digest functions (SHA1, SHA2, MD5, etc.) and cryptographic functions (Elliptic Curve Cryptography (ECC), Rivest-Shamir-Adleman Algorithm (RSA), Advanced Encryption Standard (AES), etc.). To speed up the execution of these functions, the SEs comes with an integrated crypto-processor [Remote 2014]. Typically SEs have less than 1 MB of ROM (200-500 KB) and less than 15 KB of RAM (2-5 KB public RAM, 8-10 KB of Dynamic RAM) [Urien 2015]. With these configurations, SE can store 3 to 5 applications. These applications are called codelets/applets because they are very small applications performing a very specific task.

SE is used on a large-scale in various fields like payment (chip-based credit/debit cards), communication (SIM cards), identification documents (biometric passports, chip-based ID cards), IoT [Deshpande *et al.* 2019b] (for identification, authentication, certification, verification), etc. It is also used to establish a *root of trust* in Blockchain-IoT systems [Deshpande *et al.* 2019a].

Further, there are two most popular SE types: SEs based on Multos (Multi-OS) [Multos Consortium 2020] and SEs based on Java Card [Chen 2000]. Both Multos and Java Card are Operating systems of the SEs. Applications made for Multos SE are called codelets and applications made

for Java Card SE are called applets. Both types of SEs are popular in the commercial context.

For programming a Multos SE, as it is based on security by design approach, special programming certificates called Application Load Certificate (ALC)s are needed while for Java Card SE, the same does not apply and a key is required to install/update the applets [Wilcox 2003]. This adds an additional security layer to prevent unauthorized re-programming of SE from non-owners to compromise the security of SE by installing malicious code on the SE. As Multos uses ALCs instead of a symmetric key (in case of Java SE), it is possible to reprogram multos SE remotely even in an unsecured environment.

As SEs are a critical part of security applications, they are certified under international standards of Common Criteria - Evaluation Assurance Level (CC-EAL). The certification level varies from EAL1 (lowest) to EAL7 (highest). However, there are some SEs (under EAL7+) which are said to have evaluation assurance higher than EAL7 level.

SEs can be used as a standalone secure micro-controller or in conjunction with a bigger and complex system as a Hardware Security Module (HSM). When used as HSM, SE's cryptographic tokens can be created/accessed via Public-Key Cryptography Standards (PKCS) #11 which are essentially platform independent APIs of SEs defining most commonly used cryptographic object like X.509 Certificates, RSA/DES keys, etc. Depending on the level of CC-EAL and whether or not the SE supports PKCS #11, the cost varies greatly. Also, the interfaces supported by SEs for communication further decide the cost. For communication, most SEs support Application Protocol Data Unit (APDU) ISO-7816 over serial/NFC interface. Others, although in a small minority, may support Serial Peripheral Interface (SPI) or Inter-Integrated Circuit (I2C).

In a nutshell, the SE:

- provides TEE
- has memory constraints (<1MB ROM, <20KB RAM)
- has small packaging and a crypto-processor
- is certified by CC-EAL
- may support PKCS #11, SPI, I2C

1.6 Thesis' Contributions

On a macro level, the main contribution of this thesis is the conceptualization of a system to secure *transactions* or atomic data transfer. The proposed system is perceived to have the following properties:

- Truly holistic and agile.
- Field of application agnostic i.e., very flexible.
- Modular and scalable.
- Secures both types of *transactions* i.e., data and financial.
- Provides security through TEE, TSE.
- Provides immutability through distributed storage and consensus oriented decisions.
- Responsible with non-repudiation.
- Decentralized for added reliance.

To achieve this, the conceptualized system uses two main technologies i.e. Secure Element (Section 1.5) and Blockchain (Section 1.4). This combinational approach helps to overcome the impediments of using any one technology alone. We then apply our conceptualized system to three different fields i.e., Internet of Things, Smart Vehicles and Smart Grids.

While combining these technologies under our conceptualized system, two main problems related to blockchain technology are also resolved through mathematical analysis validated by practical implementation and simulations.

The first problem is the dimensioning aspect of blockchain where currently different parameters of blockchain are arbitrarily fixed. In this thesis, we address this issue by carefully modeling blockchain's P2P network using graph theory and propose optimum parameters under different use-case scenarios.

The second problem is the problem of ascertaining the origin of data coming from blockchain to a remote resource constraint device through an intermediary. The remote device lacks the resources required for validation. We solve this problem using a novel 2 signature mechanism, applied using Secure Element technology without compromising on the efficiency or accessibility of blockchain technology for resource constrained devices.

The two above solutions related to blockchain technology can be clubbed under meso-level contributions of this thesis.

1.7 Thesis' Organization

The current version of this thesis is organized into 5 chapters which are summarized below:

Chapter 1 explains the background for the research and motivations for this thesis. It also introduces the main concepts of blockchain and SE on which the proposed system of the thesis stands.

Chapter 2 explains how blockchain can be used as a distributed database to secure transactions. Further, it illustrates the blockchain framework dimensioning (modeling) aspect as well as the evaluation aspect and explains its importance in the context of transaction security.

Chapter 3 explains the combinational approach of using blockchain and SE concurrently and applying it to the fields of Demand Response, Smart Vehicles and IoT. Various related algorithms are explained as well.

Chapter 4 explains the problem of certifying data affiliation and how SE solves it, in the context of IoT. It also deals with system liveness problem and how the SE-based PulSec framework solves it by demonstrating an Industry 4.0 use-case.

Chapter 5 concludes this thesis highlighting all important contributions of this thesis. It also elaborates on future perspectives related to the research subject of this thesis.

Chapter 2

Blockchain Modeling and Evaluation

Contents

2.1 Introduction

2.1.1 Why Blockchain to Secure Transactions?

In the new context of *transactions*, Blockchain offers an unique combination of three main properties to secure them i.e., immutability of data, persistent storage and consensus based decisions. The property of immutability is very important as once the data in a transaction is stored on the blockchain, a secured trail of transactions could be easily verified. Moreover, persistent distributed storage offered by blockchain enables to create a truly trustless environment. (Later, in Chapter 3, we explain how these properties are important to guarantee a new concept of *non-repudiable responsibility*).

Further, the property of consensus based decisions in blockchain also adds the capability to check the veracity of data within a transaction through collective decision-making, before the transaction is stored on the blockchain.

2.1.2 How to Secure Transactions using Blockchain?

Using blockchain as distributed database can surely solve many problems related to holistic security of *transactions*. In order to utilize blockchain for securing transactions, generally there are two alternatives i.e., propose a new blockchain framework or use an already existing blockchain framework. The proposition of new blockchain framework requires designing it with optimal/feasible values of parameters (dimensioning) like connectivity, diameter, block size, etc. While, using an already existing blockchain framework requires evaluation of the chosen framework for these parameters according to the target use-case.

Further, evaluation is also imperative for validating a proposed novel framework. Thus, depending on the type of framework (new or existing), modeling and evaluation of blockchain frameworks are the extremely important steps before using it as distributed database. The next two sections (Sec. 2.2 and Sec. 2.3) presents in detail, the work on how to perform modeling (dimensioning) and evaluation of blockchain framework, respectively. The results/findings of Section 2.2 for optimal/feasible blockchain parameters are then tested/evaluated for various blockchain properties in Section 2.2.5. While, Section 2.3 evaluates an existing blockchain framework Bitcoin and conclusions about its connectivity and diameter are drawn from contributions of Section 2.2.

2.2 Blockchain Framework Modeling

In blockchain technology, block and transaction messages are propagated through a distributed P2P network. Each participating peer performs local actions to create and maintain a global topology. These actions include peer discovery, neighbor selection, establishing and maintaining outbound connections, accepting or rejecting inbound connections. The blockchain P2P network properties like topology, size (maximum number of participating peers) and performance (connectivity, diameter) are highly impacted by the blockchain type, the consensus protocol used, the number of inbound/outbound connections per peer (specified by the blockchain client software) and the neighbor selection process. For example:

- Permissioned blockchains that use voting based consensus generally re-

quire a fully/highly connected P2P network between voting peers with small size (ten to hundreds peers). The topology of the clique formed by voting peers can be modeled by complete or nearly complete digraphs. A complete digraph can be considered as a special case of $(n-1)$-*out* digraphs where n is the digraph size and $n-1$ is the out-degree of each vertex.

- Permissioned blockchains that use lottery based consensus require a small size P2P networks (but larger than those used by voting based consensus) where all peers have the same small number of random outbound connections. Thus, the topology of these networks can be viewed as random r-*out* digraphs, where r is the out-degree of each vertex.
- Non-permissioned blockchains that use proof based consensus support a large number of nodes (thousands) in P2P networks where all peers have the same small number of random outbound connections. Thus, the topology of these networks can be also viewed as random r-*out* digraphs.

Therefore, many existing blockchain frameworks' P2P network can be viewed as random r-*out* digraphs. The number of outbound connections, r, is specific to each blockchain framework. Few examples are elaborated in Table 2.1. However, some important questions arise like: How is the default value of r defined? What is its impact on the blockchain performance? and what are the impacts of the in/out degree distribution, of the peer discovery protocol and of the neighbor selection process on the blockchain performance? On which, to the best of our knowledge, no deep investigation is conducted till date.

Table 2.1: Default no. of outbound connections for various blockchain frameworks.

Blockchain	r **value**
Bitcoin	8
Ethereum	13
Litecoin	125
Neuorochain	3
Stellar	8

Thus, the main objective of our work is to build an r-*out* digraph for blockchains that can be used by resource-limited devices such as IoT devices, smartphones, etc. The value of r as well as the in/out degree distribution

should be dynamically adapted depending on the desired blockchain P2P network performance in terms of connectivity, diameter and clustering. To the best of our knowledge, almost all blockchain frameworks follow random *r-out* digraphs as they specify default value of outgoing connections in their client software implementations.

However, for the case of blockchain frameworks where the underlying p2P network is not *r-out* graph, our work can be adapted accordingly, by selecting an appropriate graph type and generator and using it in the secure overlay layer for the topology construction as explained in the next sections of this chapter. Next, we consider the key properties as elaborated in Section 1.4.1 for enhancing the blockchain P2P network performance.

2.2.1 Generating r-*out* Digraphs

The key topics mentioned in Section 1.4.1 are important to optimize blockchain topology w.r.t a given use-case scenario. These key topics are therefore to be modeled using a structure close to the actual P2P network. For this, we use graph theory to model blockchain's P2P network. In the context of blockchain, the underlying P2P network construction starts by finding peers with a peer discovery method. For peer discovery, distributed or centralized methods can be applied. With distributed method, each peer performs neighbor discovery independently using different techniques (e.g., asking default servers for the neighbor list, listening to address advertisement messages, using default neighbor addresses) [bitcoin.it]. Contrarily, with the centralized method, each peer registers on the centralized server(s), who alone has the vision of all online peers.

Next, we consider a special class of graph called *r-out* digraph for modeling the P2P network. Our motivation comes from the fact that most blockchains' P2P network, at any given time, maintain r outbound connections. Further, we consider *r-out* graph generation techniques for both settings i.e. centralized as well as distributed, to incorporate peer discovery properties. The *r-out* digraphs can likewise be generated in two ways i.e. centralized and distributed.

2.2.1.1 Distributed r-*out* Graph Generation

The distributed method of *r-out* graph generation is the most commonly used in blockchain architectures. In this type of generation, each individual node selects its own r peer neighbors. The main advantage for such type of generation is that it is very simple without the involvement of any centralized entity. Due to this, such distributed *r-out* graph generation are used in almost all non-permissioned/public blockchain.

Further, the distributed method is more resilient against failures of other peers but is more resource intensive. The resulting blockchain architecture is immune to attacks like DDoS because, in a non-permissioned/public blockchain, any node can participate in without a centralized access control and the graph generation is distributed. This makes distributed *r-out* graph generation the default choice as higher resiliency results in higher reliance on the financial transactions from the blockchain.

However, there are certain disadvantages associated with the distributed *r-out* graph generation like difficulty in controlling topology. For e.g., in an open and unregulated network, clusters are bound to form around the nodes that are more active and with more resources. However, as the topology generation/graph is distributed, any controlling action results in a very high overhead. Another issue with the distributed *r-out* graph generation is the resources overhead related to the neighbor discovery protocol.

As there is no centralized database, the participants periodically transmit the messages of the new peer participants so as to keep the network as uniform as possible when new connections are added. Further, as the network participation is non-permissioned in most blockchain frameworks that utilize distributed method, it is more difficult to verify data transactions as the owner cannot be effectively verified.

2.2.1.2 Centralized r-*out* Graph Generation

The peer discovery using centralized *r-out* graph generation is fast without requiring additional resources (bandwidth, power consumption and storage of all neighbors information, etc.). In the context of IoT, this is advantageous for mobile peers and remote nodes with limited resources. However, this protocol is prone to Denial of Service (DoS) attacks and is less resilient against the failure of the centralized server (Single Point of Failure (SPoF)). Hence, a group of backup servers are necessary. This allows us to mitigate SPoF and DoS problems.

Topology control is easier with the centralized approach than with the distributed approach for *r-out* graph generation as each peer has a direct communication link with the centralized server(s) and the centralized server(s) can quickly detect when a node leaves/joins the network. As a result, the topology reconstruction is fast and efficient. As centralized server(s) is/are incharge of the topology control, it further adds to the flexibility of the network and on-demand required changes to manage temporary surge can be effectively and quickly implemented.

Although, it introduces hierarchy in the topology management, the blockchain functions still remains decentralized. Another advantage of the

centralized method is that since all the node addition/removal goes through centralized server(s), the identity of each node can be ascertained using technologies like SE [GlobalPlatform, Inc. 2018]. SE has been already to guarantee *root of trust* in Industry 4.0 [Deshpande *et al.* 2019b], IoT [Deshpande *et al.* 2019a], Smart Vehicles [Deshpande *et al.* 2019c], etc. SE has also been used in blockchain transaction security [Urien 2018] which was implemented in the context of marketplace transactions in Demand Response [Deshpande *et al.* 2020]. In our work, we propose to use SE and extend its application to include the functionality of authentication of lower layer peers, verifying data transactions and overall maintain trust in the blockchain network.

2.2.1.3 Selecting the Appropriate Graph Generation Technique

A neighbor selection process may result in two types of P2P networks i.e., structured and unstructured. In an unstructured P2P network, the topology is not optimized with respect to parameters like neighbor-degree, network overlay, physical position of peers and metrics like bandwidth, link-latency, computing and storage capacities of peers [Beverly & Afergan 2007]. Moreover, network properties like connectivity, small diameter, specific clustering coefficient, etc., cannot be guaranteed. Hence, in our work, we focus on a structured P2P network to optimize the topology according to given network properties specifications.

As constructing an optimized topology structure using distributed protocols is an arduous task, centralized protocols are preferred [Srivatsa *et al.* 2006] which is also the point of view of our work. For a given P2P network structure, there can be different centralized graph generators and selecting a simple, fast and robust generator is of prime importance. In our work, we consider two types of r-*out* graphs i.e. approximate (average r outbound connections per node) and exact (exactly r outbound connections per node).

2.2.2 Centralized Topology Management and Maintenance

In this Section, we describe the general architecture of our proposed 2-layer network where the upper layer helps to control the topology of the lower layer in a centralized manner. The lower layer forms the P2P network of the blockchain and implements all the blockchain protocols and their functions except topology control protocols (neighbor discovery, selection and maintenance).

2.2.2.1 General Architecture

Our hybrid P2P network is organized into two layers: the upper tier - servers and the lower tier - peers (Figure 2.1). In our work, we consider that the blockchain is built on the lower layer of a hybrid P2P network infrastructure deployed over the Internet. The infrastructure is formed by peers and servers. The peer devices can be mobile IoT devices like delivery drones, in-vehicle electronic control units, survey robots, etc., or static IoT devices like weather stations, automatic toll plazas, disaster alerting systems, embedded devices (for various applications), etc.

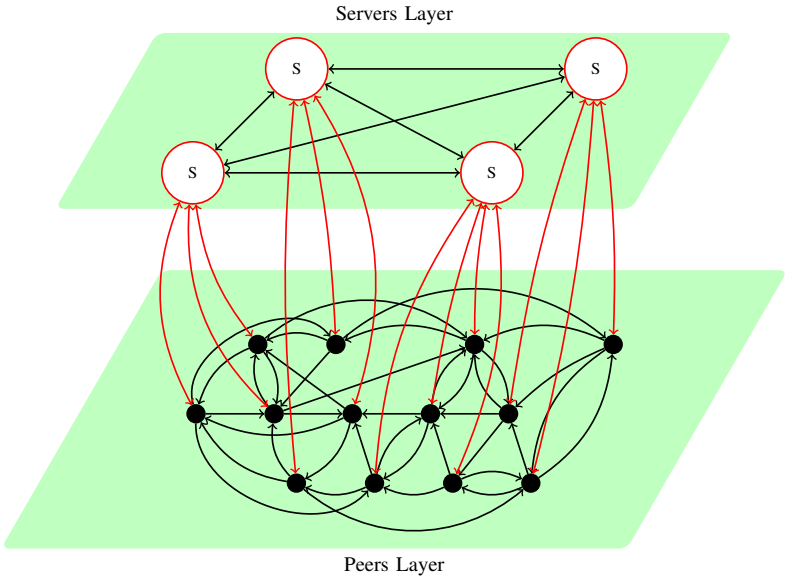


Figure 2.1: General architecture of the two layers.

Each peer has the capability to communicate with other peers that are indicated as neighbors by upper layer servers. The servers' role is to collect information from peers and elect a designated server responsible for selecting and maintaining outbound and inbound neighbors of all peers according to their profiles. We distinguish two types of profiles for peers: contributors and non-contributors. A contributor can be seen as router who allows information forwarding by accepting inbound connections. Non-contributor does not accept inbound connections. Once a peer has information on its neighbors through its local server, it can directly communicate with them.

All communications are done over TCP/IP protocols. Each peer is identified by its IP address and should initiate outbound connections with other peers. Only contributor peers accept inbound connections. To allow inbound connections for contributor peers belonging to private networks, different ways can be applied:

- Setting up explicit TCP port forwarding rules on the contributor (router).
- Using Network Address Translation (NAT) traversal solutions like: Relaying via Rendezvous Servers, Connection Reversal and TCP Hole Punching [Yu & Shao-Hai 2006].

This “2-tier architecture" is helpful to satisfy the constraints on full connectivity, easy topology control and increases network flexibility. As the upper layer contains multiple servers acting as mirrors, a failure in the designated server can be resolved by re-selecting a new server from the server pool.

2.2.2.2 Upper Layer Operations

The principal upper layer operations are used to maintain the described architecture and to control the topology of the lower layer.

2.2.2.2.1 P2P Network Topology Control

The role of this task is to discover online peers, build and maintain the lower layer P2P network topology. In peer discovery, when a peer is online, it first does the authentication with the associated server (using SE). Then it indicates its available resources (computing power, storage, battery autonomy) and its profile: contributor (allows inbound connections) or non-contributor (no inbound connections). The associated server then forwards the collected information to the designated server.

Based on the information provided by peers, the designated server builds the topology using a graph generator. To this end, it pseudo-randomly selects inbound and outbound neighbors for each contributor and only outbound neighbors for non-contributor. When a new peer joins or when an old peer leaves the network, designated server updates the network topology accordingly.

2.2.2.2.2 Server-Pool Maintenance

This task consists of (pseudo-randomly) selecting a designated server from the server-pool and maintaining the synchronized view of the topology of the lower layer across server-pool. The server-pool consists of regional servers which serve a specific region independently. The regional server is acting as an intermediary between regional peers and the designated server. It is responsible for collecting information from regional peers, forwarding it to the

designated server and finally informing its regional peers about their neighbors. If any server (designated or regional) fails, another is selected from the server-pool as a replacement.

2.2.2.3 Lower Layer Operations

In this layer, peers are equal and follow a flat hierarchy. However, we distinguish them in two types: contributors (accepting inbound connections) and non-contributors (rejecting inbound connections). The whole blockchain resides in this layer in true sense. As a result, there are no additional security implications arising out of 2-layer structure. Contributing peers support blockchain database, routing, mining and are a source of new information. This makes them indispensable for proper blockchain functioning. Non-contributing peers, on the other hand, are the only source of new information and don't contribute for blockchain's up-keep and maintenance. The functionality of peer discovery and peer selection are delegated to the upper layer.

Note: The proposed solution of two layers does not affect the distributed properties of the blockchain P2P network. This is because the P2P network generation is still random and based on the whole network knowledge. In other words, for a given peer, its neighbors are selected randomly among all the network participants and not from a subset/partial network view, unlike the current blockchain P2P networks. In addition, with our proposed solution, the overlay layer can be queried, to select neighbors for a given node from a small subset of randomly selected nodes. This would make the constructed topology more closer to the existing blockchain topology generation techniques.

Further, we recall that the blockchain operations(block proposal, transactions and block propagation, block validation by consensus algorithms) are still performed in the distributed P2P network of the nodes. Only the P2P network topology generation and maintenance operations are ensured by the secure overlay layer to remove the excess network topology control/maintenance overhead and increase configurability.

2.2.3 Blockchain P2P Network Modeling

In this Section, we model the P2P network described in the Section 2.2.2 by a special class of digraphs, named *r-out* digraphs. We discuss three models to generate such random digraphs. Further, we suppose that the P2P network we model has n peers. Among them, k non-contributor peers exist having *null* inbound connections. The other peers are contributors and accept inbound

connections up to a limit γ. Each peer, regardless of its profile, initiates and maintains r outbound connections.

2.2.3.1 Definitions and Notations

We use the following notations and definitions: Let $D = (V, \overrightarrow{E})$ be a simple labeled digraph (directed graph), where $V = \{v_1, v_2, \ldots, v_n\}$ is the set distinguishable vertices of size n, and $\overrightarrow{E} = \{(u, v) : u, v \in V\}$ is the set of directed edges. The digraph D is called simple if it does not contain self-loops (no directed edge between a vertex and itself) or multiple directed edges in the same direction. The set of all in-neighbors and out-neighbors of a vertex u are denoted by $N_D^-(u)$ and $N_D^+(u)$, respectively. Their cardinalities are the in-degree and the out-degree of u, denoted by $deg_D^-(u)$ and $deg_D^+(u)$, respectively. The sum of out-degree and in-degree of u gives its degree, denoted by $deg_D(u)$. The maximum in-degree of D, denoted by $\Delta^-(D)$, is the largest in-degree of its vertices. We define the degree sequence of D as the list of its in-degree and out-degree pairs, i.e.,

$$\big((deg_D^+(v_1), deg_D^-(v_1)), \ldots, (deg_D^+(v_n), deg_D^-(v_n))\big)$$

Since each directed edge in a digraph D increases out-degree of its head and in-degree of its tail by one, we have

$$\sum_{u \in V} deg_D^+(u) = \sum_{u \in V} deg_D^-(u) = \operatorname{abs} \overrightarrow{E}. \tag{2.1}$$

A digraph, D, is called *r-out* digraph if each vertex in V has the same out-degree r. More formally: $\forall u \in V : deg_D^+(u) = r$. Its total number of arcs is indicated by the following Lemma:

Lemma 1. *If D is an r-out digraph then,* $\operatorname{abs} \overrightarrow{E} = rn$.

Proof. Let $D = (V, \overrightarrow{E})$ be a simple *r-out* digraph of order n. Based on equation 2.1, we have:

$$\operatorname{abs} \overrightarrow{E} = \sum_{u \in V} deg_D^+(u) = rn.$$

□

2.2.3.2 Model Description and Motivation

We model the lower layer P2P network by a simple *r-out* digraph. This model is motivated by the fact that it is sufficient to set $r = 2$ to obtain with high probability a fully connected network as its size grows to infinity

[Mauldin 1982]. Consequently, the full connectivity constraint can be satisfied. Also, the justification for selecting a digraph instead of an undirected graph is that connections in our blockchain protocol can be inbound or outbound, based on who initiated the connection. Since each peer should maintain exactly r outbound connections to distinct target peers, the resulting P2P network topology can be represented as an r-*out* digraph. Further, only single inbound and/or outbound connection between two peers is supported, a simple r-*out* digraph satisfies this restriction.

In order to consider non-contributor, we assume that the simple r-*out* digraph contains k, $0 \leq k \leq n-2$, vertices with null in-degree. All the other vertices, representing contributor peers, have up to γ inbound degrees which correspond to the maximum number of inbound neighbors. We denote such digraph by $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$. The limits on outbound and inbound degrees are mainly to prevent excessive bandwidth occupation.

2.2.3.3 Satisfying Necessary Conditions of Feasibility

To generate a simple r-*out* digraph $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$, some necessary (but not sufficient) conditions on its parameters n, k, r, γ needs to be satisfied for its realization. All the parameters are correlated. In general, n and k are fixed according to the use-case for which the blockchain is used. Based on n and k, range for parameters r and γ are derived and expressed.

For parameters n and k, it is obvious that $n \geq 2$ to create a network of at least 2 participating peers and $0 \leq k \leq n-2$ to have at least 1-*out* digraph. For the parameter r, the total number of directed edges of a r-*out* digraph should be less than or equal to total possible edges of a complete digraph for same value of n and k. As the total number of directed edges of a r-*out* digraph is $r \times n$ (from Lemma 1) and the total possible edges of a complete digraph with n nodes and k non-contributors is $2\binom{n}{2} - k(n-1)$, we find that:

$$r \leq \frac{(n-1) \times (n-k)}{n} \tag{2.2}$$

For γ, let $d = ((r, r_1), (r, r_2), \ldots, (r, r_n))$ be the degree sequence of digraph $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$, where $r = deg^{+}_{\mathbb{D}^{\gamma}_{n,k,r\text{-}out}}(v_i)$ and $r_i = deg^{-}_{\mathbb{D}^{\gamma}_{n,k,r\text{-}out}}(v_i)$ for $v_i \in V(\mathbb{D}^{\gamma}_{n,k,r\text{-}out})$ and $i = 1, \ldots, n$. Our objective is to find the optimized range of γ satisfying $r_i \leq \gamma$ for $i = 1, \ldots, n$. The following Lemma gives such range.

Lemma 2. *$\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ is realizable if the inbound degree limit γ satisfies $\frac{rn}{n-k} \leq \gamma \leq n-1$.*

Proof. As the maximum number of in-neighbors of a contributor peer is $n-1$ (excluding itself), the highest value of γ, $\gamma_{\max}$, is $n-1$. On the other hand,

the lowest value of γ, $\gamma_{\min}$, is reached when all the contributors have the same inbound degree. Consequently, $\gamma_{\min}$ can be written as $\sum_{i=1}^{n-k} \gamma_{\min} = r \times n$ and therefore, $\gamma_{\min} = \frac{rn}{n-k}$. □

2.2.3.4 Generating Simple Random r-*out* Digraphs

A simple random *r-out* digraph $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ is a digraph sampled out from a set of all *r-out* digraphs ($\mathscr{D}^{\gamma}_{n,k,r\text{-}out}$) according to the uniform distribution. For a given vertex, other probability distributions can also be used to fine tune desired properties, like clustering, regularity, smaller diameter, etc. We investigate three models: Binomial, Uniform and Preferential Attachment. Exact simple *r-out* digraphs can be generated by Binomial and Preferential Attachment models whereas approximate simple *r-out* digraphs (mean out-degree is r) can be generated by Binomial and Uniform models. The rationale behind investigating approximate simple *r-out* digraphs is to show if it is possible to use them to keep the exact *r-out* digraph properties and speed up the topology construction.

2.2.3.4.1 Approximate *r-out* Digraph Generation Using Binomial Model

A binomial model has two parameters, the number of vertices n and the probability p ($0 \leq p \leq 1$) assigned independently to each of $2\binom{n}{2}$ possible directed edge. Let $\mathbb{D}_{n,p}$ be a binomial random digraph. $\mathbb{D}_{n,p}$ is considered as an approximate *r-out* digraph $\mathbb{D}_{n,k,r\text{-}out}$ if the average out-degree of its vertices is r and the number of vertices having null in-degree is k. Let us consider $\mathbb{D}_{n,k,p}$ as an approximation of $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$. The average number of directed edges of $\mathbb{D}_{n,k,p}$, $\text{abs}\,\overrightarrow{E}(\mathbb{D}_{n,k,p})$, should be equal to the number of edges of $\mathbb{D}_{n,k,r\text{-}out}$, $\text{abs}\,\overrightarrow{E}(\mathbb{D}^{\gamma}_{n,k,r\text{-}out})$. To ensure this equality, the probability p is given in the following Lemma:

Lemma 3. *The equality* $\text{abs}\,\overrightarrow{E}(\mathbb{D}_{n,k,p}) = \text{abs}\,\overrightarrow{E}(\mathbb{D}_{n,k,r\text{-}out})$ *holds true when* $p = \frac{rn}{(n-1)(n-k)}$.

Proof. As the total number of directed edges of complete digraph on n nodes and k as non-contributors among them is $2\binom{n}{2} - k(n-1)$, the average number of directed edges of $\mathbb{D}_{n,k,p}$ is: $\text{abs}\,\overrightarrow{E}(\mathbb{D}_{n,k,p}) = \left(2\binom{n}{2} - k(n-1)\right) \times p$. We also

have abs $\overrightarrow{E}(\mathbb{D}_{n,k,r\text{-}out}) = r \times n$ (from Lemma 1). By resolving the equality, the probability p can be expressed as: $p = \frac{rn}{(n-1)(n-k)}$ □

As the average out-degree of nodes is $p \times (n-k)$, the probability $p = \frac{rn}{(n-1)(n-k)}$ ensures that the average out-degree of nodes tends to r when the number of nodes n goes to ∞. Thus, $\mathbb{D}_{n,k,p}$ sustains the concept of r-*out* digraph at average.

Let D_0 be a given digraph generated by the model $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$. An approximate digraph of D_0 can be generated by the $\mathbb{D}_{n,k,p}$ model with the following probability:

$$\mathbb{P}(\mathbb{D}_{n,k,p} = D_0) = p^{r\times n} \times (1-p)^{(n-1)(n-k)-rn}. \tag{2.3}$$

The generation process of an approximate random digraph $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ using a $\mathbb{D}_{n,k,p}$ model consists of multiple iterations. Starting with an empty digraph of n nodes and *zero* arcs, performing $2\begin{pmatrix} n \\ 2 \end{pmatrix} - k(n-1)$ independent Bernoulli trails to add arcs each with probability $p = \frac{rn}{(n-1)(n-k)}$.

Based on this model, the resulting P2P network may not follow the fixed input parameters (r, k, γ). It may have some peers with more/less r outbound connections or more than γ inbound connections and non-contributors accepting inbound connections. If we accept this network topology, we don't strictly enforce the fixed parameters. However, it benefits from the r-*out* digraph properties. The following Lemma shows that each of the r-*out* digraphs can be generated with same probability when using binomial random digraph generator with out-degree constraints.

Lemma 4. *A random digraph $\mathbb{D}_{n,p}$ subject to the constraint that each node has exactly the same out-degree r, is equally likely to be one of the $\binom{n-1}{r}^n$ digraphs of $\mathscr{R}$-out.*

Proof. Let D_0 be an r-*out* digraph. Based on the remark that $\{\mathbb{D}_{n,p} = D_0\} \subseteq \{\mathbb{D}_{n,p} \in \mathscr{R}\text{-out}\}$, we have

$$\begin{aligned}
\mathbb{P}(\mathbb{D}_{n,p} = D_0 | \mathbb{D}_{n,p} \in \mathscr{R}\text{-out}) &= \frac{\mathbb{P}(\mathbb{D}_{n,p} = D_0)}{\mathbb{P}(\mathbb{D}_{n,p} \in \mathscr{R}\text{-out})} \\
&= \frac{\left(p^r \times p^{(n-1)-r}\right)^n}{\left(\binom{n-1}{r} p^r \times p^{(n-1)-r}\right)^n} \\
&= \binom{n-1}{r}^{-n}
\end{aligned}$$

□

2.2.3.4.2 Approximate *r-out* Digraph Generation Using Uniform Model

This graph model has two parameters, the number of vertices n and the number of directed edges m, where m takes values from zero to the maximum number of directed edges derived from complete simple digraphs. The digraph also contains k vertices with *zero* in-degree. Therefore, the maximum number of directed edges are $2\begin{pmatrix} n \\ 2 \end{pmatrix} - k(n-1)$. This can be simplified further into the form $(n-1)(n-k)$. Therefore, the number of directed edges m is within the interval $\{0, (n-1)(n-k)\}$. Let $\mathscr{D}_{n,m}$ be the family of all possible digraphs having n nodes and exactly m directed edges, and $\mathbb{D}_{n,m}$ be a random digraph chosen uniformly at random from $\mathscr{D}_{n,m}$. The cardinality of $\mathscr{D}_{n,m}$ is $\operatorname{abs}\mathscr{D}_{n,m} = \begin{pmatrix} 2\binom{n}{2} \\ m \end{pmatrix}$. Each digraph $D \in \mathscr{D}_{n,m}$ is chosen uniformly with equal probability:

$$\mathbb{P}(\mathbb{D}_{n,m} = D) = \begin{cases} \dfrac{1}{\begin{pmatrix} 2\binom{n}{2} \\ m \end{pmatrix}} & \text{if } \operatorname{abs}\overrightarrow{E} = m \\ 0 & \text{if } \operatorname{abs}\overrightarrow{E} \neq m \end{cases} \tag{2.4}$$

The digraph generation process consists of multiple iterations, starting with a graph of n vertices and *zero* directed edges, adding arcs till total edges are m thereby the resulting graph is equally likely to be one of the $\mathscr{D}_{n,m}$ set.

From Lemma 1, we know that an *r-out* digraph has exactly $r \times n$ directed edges. By taking $m = rn$, the set of all *r-out* digraphs can be sampled using the uniform random directed graph $\mathbb{D}_{n,rn}$. The cardinality of this set can be given by the following Lemma:

Lemma 5. *Let $\mathscr{R}$-out be the family of all possible r-out digraphs. Then,* $\operatorname{abs}\mathscr{R}\text{-}out = \begin{pmatrix} n-1 \\ r \end{pmatrix}^n$

Proof. For a given vertex, the number of ways to choose r out-neighbors from $n-1$ possible candidate is given by $\begin{pmatrix} n-1 \\ r \end{pmatrix}$. As the out-neighbor sets of all vertices(n) are mutually independent, the total number of all possible *r-out* digraphs can be given by $\begin{pmatrix} n-1 \\ r \end{pmatrix}^n$. □

Based on Lemma 5 and equation 2.4, the random digraph $\mathbb{D}_{n,r\times n}$ can be

any r-*out* digraph with the following probability:

$$\mathbb{P}(\mathbb{D}_{n,rn} \in \mathscr{R}\text{-out}) = \frac{\operatorname{abs} \mathscr{R}\text{-out}}{\operatorname{abs} \mathscr{D}_{n,rn}} = \frac{\binom{n-1}{r}^n}{\binom{2\binom{n}{2}}{rn}} \tag{2.5}$$

The following Lemma shows that each r-*out* digraphs can be generated with same probability when using uniform random digraph generators with out-degree constraints.

Lemma 6. *A random digraph $\mathbb{D}_{n,r\times n}$ subject to the constraint that each node has exactly the same out-degree r, is equally likely to be one of the $\binom{n-1}{r}^n$ digraphs of $\mathscr{R}$-out.*

Proof. Let D_0 be an r-*out* digraph. Based on the remark that $\{\mathbb{D}_{n,rn} = D_0\} \subseteq \{\mathbb{D}_{n,rn} \in \mathscr{R}\text{-out}\}$, we have

$$\begin{aligned}
\mathbb{P}(\mathbb{D}_{n,r\times n} = D_0 | \mathbb{D}_{n,r\times n} \in \mathscr{R}\text{-out}) &= \frac{\mathbb{P}(\mathbb{D}_{n,rn} = D_0)}{\mathbb{P}(\mathbb{D}_{n,rn} \in \mathscr{R}\text{-out})} \\
&= \frac{\binom{2\binom{n}{2}}{rn}^{-1}}{\frac{\binom{n-1}{r}^n}{\binom{2.\binom{n}{2}}{rn}}} \\
&= \binom{n-1}{r}^{-n}
\end{aligned}$$

□

2.2.3.4.3 Observations

Our objective is to find a simple random model generator to sample r-*out* digraphs. From Lemmas 6 and 4, we can see that both $\mathbb{D}_{n,r\times n}$ and $\mathbb{D}_{n,p}$ under the constraint that all vertices have the same out-degree (r), are equally probable in distribution and the resulting r-*out* digraph is selected uniformly at random from $\mathscr{R}$-out (the set of all r-*out* digraphs). In other words, uniform and binomial models subject to the above constraint, can generate all

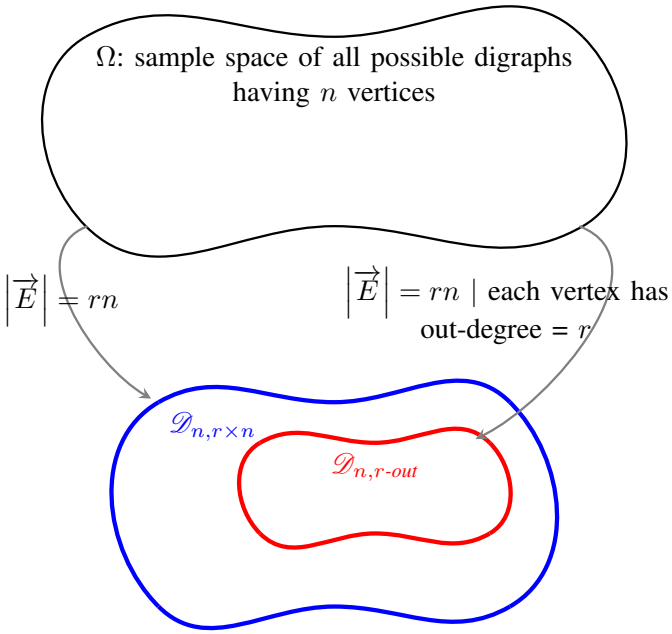


Figure 2.2: Sample spaces for approximate digraphs.

possible r-*out* digraphs, each being equally likely. Figure 2.2 illustrates such generation mechanisms from the sample space of all possible digraphs on n vertices. We know from random graph theory that random digraphs $\mathbb{D}_{n,rn}$ and $\mathbb{D}_{n,p}$ have the same asymptotic behavior for most of properties when the mean number of arcs in $\mathbb{D}_{n,p}$ is equal or close to $r \times n$ [Frieze & Karoński 2016]. In other words, when $2\binom{n}{2}p = r \times n$ or $p = \frac{r}{n-1}$. In practice, $\mathbb{D}_{n,\frac{r}{n-1}}$ is easier to handle then $\mathbb{D}_{n,rn}$, i.e. approximate r-*out* graph generation using binomial model is computationally easy compared to its uniform model counterpart.

Based on the above analysis, the binomial random digraph model $\mathbb{D}_{n,p=\frac{r}{n-1}}$ subject to the constraint that all vertices have the same out-degree (r) is selected to sample our r-*out* digraphs rather than the uniform random digraph model $\mathbb{D}_{n,rn}$.

2.2.3.4.4 Exact r-*out* Digraph Generation Using Binomial Model

To generate an exact r-*out* digraph $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ by the $\mathbb{D}_{n,k,p}$ model, additional constraints are necessary. When a Bernoulli trial results in adding a directed edge between pair of vertices (the first vertex is called tail while the second one is called head), three additional independent conditions should be filled: a) the tail is a contributor, b) the tail has less than r out-neighbors and c) the head has less than γ in-neighbors. Consequently, the probability to add a directed

edge is a result of multiplying four probabilities: p, $\mathbb{P}(\text{tail} = \text{contributor})$, $\mathbb{P}(deg^+_D(\text{tail}) < r)$ and $\mathbb{P}(deg^-_D(\text{head}) < \gamma)$. The probability p is given by $\mathbb{D}_{n,k,p}$ model as $\frac{rn}{(n-1)(n-k)}$ (from Lemma 3). The probability for a vertex to be contributor is $\frac{n-k}{n}$. As the degree distribution is binomial, the probabilities $\mathbb{P}(deg^+_D(\text{tail}) < r)$ and $\mathbb{P}(deg^-_D(\text{head}) < \gamma)$ can be given respectively by $\sum_{i=0}^{r-1} \binom{n-k}{i} p^i (1-p)^{(n-k-i)}$ and $\sum_{i=0}^{\gamma-1} \binom{n-1}{i} p^i (1-p)^{(n-k-i)}$.

Let D_0 be a given digraph generated by the model $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$. The probability to sample D_0 using the $\mathbb{D}_{n,k,p}$ model with the additional constraints can be written as:

$$\begin{aligned}\mathbb{P}(\mathbb{D}_{n,k,p} = D_0) = \Bigg(& p \times \frac{n-k}{n} \times \sum_{i=0}^{r-1} \binom{n-k}{i} p^i (1-p)^{(n-k-i)} \times \\ & \sum_{i=0}^{\gamma-1} \binom{n-1}{i} p^i (1-p)^{(n-k-i)} \Bigg)^{r \times n} \times \Bigg(1 - \Bigg[p \times \\ & \frac{n-k}{n} \times \sum_{i=0}^{r-1} \binom{n-k}{i} p^i (1-p)^{(n-k-i)} \times \\ & \sum_{i=0}^{\gamma-1} \binom{n-1}{i} p^i (1-p)^{(n-k-i)} \Bigg] \Bigg)^{(n-1)(n-k)-rn} .\end{aligned} \tag{2.6}$$

Based on equations 2.3 and 2.6, we can see that, at average, the probability to generate an exact *r-out* digraph is less than the probability to generate an approximate one. In other words, at average, generation of an approximate *r-out* digraph is quicker compared to its exact counterpart (see Section 2.2.5.5). Consequently, the generation process for an exact $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ using the $\mathbb{D}_{n,k,p}$ model needs *at least* (best case scenario) the same number of iterations as for generating the approximate $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$. However, these iterations can largely be reduced by avoiding vertices whose in/out degrees are saturated w.r.t threshold. Subsequently, generation time is greatly minimized (see Section 2.2.5.5). Another advantage of this model is that the P2P network inherits all the properties of $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraphs.

2.2.3.4.5 Exact *r-out* Digraph Generation Using Preferential Attachment

In random digraphs, preferential attachment refers to the process of constructing a digraph by adding oriented edges, one at a time. At each step, higher degree vertices are more likely to be incident to the next selected oriented edge. This mechanism can be applied to generate random $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraphs. For this, the candidate tail and head vertices should also satisfy the

conditions on out-degree and in-degree respectively. The proposed algorithm is illustrated below:

Algorithm 1 Preferential Attachment Random $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ Digraph Generation.

Output: Random $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraph;

Initialization

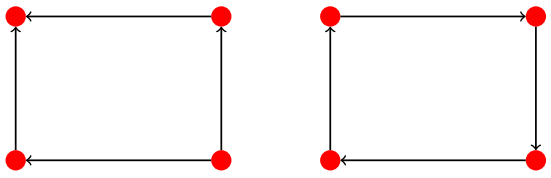
Create an empty digraph D with n vertices and k non-contributors;
Each vertex i in digraph D is assigned an initial weight α_i;

LOOP Process

while there exists vertices in D with out-degree $< r$ **do**
  Select a vertex, u, having $deg^{+}_{D}(u) < r$, uniformly at random from D;
  if there exists vertices in D with in-degree $< \gamma$ **then**
    Select a vertex, v, having $deg^{-}_{D}(v) < \gamma$ **and** maximizing the probability: $\frac{\alpha_v}{\sum_{x=1}^{n} \alpha_x}$ **and** $v \notin k$;
    Form an edge $u \rightarrow v$ in D;
    Increment the weight of v: $\alpha_v = \alpha_v + 1$;
  else
    Digraph D unrealisable with current *seed* value;
    Exit;
$\mathbb{D}^{\gamma}_{n,k,r\text{-}out} = D$;
return $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$;

The resulting random $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraph's distribution depends on the initial weight vector $\alpha = (\alpha_0, \alpha_1, \ldots, \alpha_n)$. Further, when the elements of the vector α are same and high (tends to ∞), the heads selection follow a random uniform distribution. As the tails selection also follows a uniform distribution, all the sampled digraphs, in this case, have the same uniform distribution. In contrast, when the elements of the vector α are same and low (tends to 0), head nodes are more likely selected as their in-degree increases (tends to γ) and bigger clusters appear more frequently. Further, a vertex with higher α value is more likely to be head of a higher number of peers (at most γ peers).

The initial weights given to each peer should be motivated by the P2P network topology control. When P2P network has some peers with more resources (e.g., miners, full nodes), they are more likely to have the capacity to handle higher in-degree. Such peers can be modeled by corresponding higher initial weight. When peers have equal resources, higher and equal initial weights are more appropriate to generate a uniform digraph.



(a) Weakly connected digraph

(b) Strongly connected digraph

Figure 2.3: Connectivity in digraph.

2.2.4 Random r-*out* Digraph Properties

In this Section, we explain the following properties which play an important role in information dissemination and optimization of a blockchain P2P network.

2.2.4.1 Network Connectivity

We distinguish between two types of connectivity in digraphs: weak and strong. A digraph is said to be weakly connected if its underlying undirected graph is connected. An underlying undirected graph is obtained by replacing each directed edge by an undirected edge. A digraph is said to be strongly connected if every vertex is reachable from every other vertex, i.e. there exist a path in each direction for each pair of vertices in the digraph.

In our work, we focus only on *weak connectivity* because in blockchain P2P network, even though the protocols make a distinction between inbound/outbound connections (based on who initiated the connection), the information flow on both types of connection is bidirectional. This makes the P2P network undirected when information flow is considered. As we consider two types of r-*out* digraphs i.e., approximate and exact, the connectivity constraints are different for both.

2.2.4.1.1 Weak Connectivity of Exact r-*out* Digraphs

For weak connectivity of exact r-*out* digraph, we use the theorem shown in [Mauldin 1982] which states that:

$$\lim_{n\to\infty} \mathbb{P}(r\text{-}out \text{ } digraph \text{ } is \text{ } weakly \text{ } connected) = \begin{cases} 0 & \text{if } r = 1 \\ 1 & \text{if } r \geq 2 \end{cases} \tag{2.7}$$

In other words, an r-*out* digraph is weakly connected for $r \geq 2$ with a very high probability when the number of nodes is high. This theorem is valid for

all r-*out* digraphs and hence it can be applied to our $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraph. This greatly simplifies the connectivity constraint on the exact r-*out* P2P network.

2.2.4.1.2 Weak Connectivity of Approximate r-*out* Digraphs

W.r.t graph theory, as these digraphs are an approximation of exact r-*out* digraphs, the theorem of weak connectivity for exact r-*out* digraphs cannot be directly applied. Hence, to derive the conditions of weak connectivity for approximate r-*out* digraphs, we use connectivity conditions of binomial random graphs. By ignoring directed edges from a random $\mathbb{D}_{n,k,p}$ digraph, we obtain an undirected underlying random graph $\mathbb{G}_{n,k,p'}$ where the probability p' is computed based on p. This observation is illustrated by the following Lemma:

Lemma 7. *The underlying random graph of* $\mathbb{D}_{n,k,p=\frac{rn}{(n-1)(n-k)}}$ *is* $\mathbb{G}_{n,k,p'=p(2-p)}$.

Proof. Let u and v two vertices from $V(\mathbb{D}_{n,k,p=\frac{rn}{(n-1)(n-k)}})$. We consider three events: e=$\{$there is an undirected edge between u and $v\}$, e_1=$\{$there is a directed edge from u to $v\}$, and e_2=$\{$there is a directed edge from v to $u\}$. We know from $\mathbb{D}_{n,k,p=\frac{rn}{(n-1)(n-k)}}$ that the events e_1 and e_2 are independent and $\mathbb{P}(e_1) = \mathbb{P}(e_2) = p$. The event e can be written as $e_1 \cup e_2$. Therefore:

$$\begin{aligned}\mathbb{P}(e) = \mathbb{P}(e_1 \cup e_2) &= \mathbb{P}(e_1) + \mathbb{P}(e_1) - \mathbb{P}(e_1)\mathbb{P}(e_2) \\ &= 2p - p^2 = p(2-p)\end{aligned}$$

By applying the previous analysis for all pairs of vertices, we find that they are assigned the same probability value $p' = p(2-p)$. As undirected edges are independent, the resulting $\mathbb{G}_{n,k,p'=p(2-p)}$ graph has a binomial distribution. □

We know from connectivity of random binomial graphs, that the function $\frac{\ln(n)}{n}$ is a threshold function for the disappearance of isolated vertices and so for connectivity in $\mathbb{G}_{n,k,p'}$. In other words, $\mathbb{G}_{n,p'}$ is connected if:

$$p' > \frac{\ln(n)}{n}. \tag{2.8}$$

By replacing p' with $p(2-p)$ (from Lemma 7) and p with $\frac{rn}{(n-1)(n-k)}$ (from Lemma 3), the inequality 2.8 becomes:

$$\frac{n^2}{(n-1)^2(n-k)^2}r^2 - \frac{2n}{(n-1)(n-k)}r + \frac{\ln(n)}{n} < 0. \tag{2.9}$$

The range of possible solutions of inequality 2.9 which satisfy the conditions of feasibility can be expressed as:

$$r_{\mathrm{c}} = \frac{(n-1)(n-k)}{n}(1 - \sqrt{1 - \ln(n)/n}) < r \leq n - 1 = r_{\max}. \tag{2.10}$$

The above equation is important as it establishes a relation between connectivity and the number of outbound connections r. This relation helps to model a connected blockchain P2P network with minimum possible outbound connections r_c (hence minimum resources).

2.2.4.2 Diameter

The diameter of a graph/digraph is the smallest-longest path between the vertices i.e., the maximum eccentricity of any vertex in the graph or the greatest distance between any pair of vertices in the graph. For optimally modeled blockchain P2P network, smaller diameter helps reducing information dissemination time. W.r.t diameter, exact and approximate r-*out* digraphs have the same asymptotic upper bound $O(\log n)$ [Barzdin 1973, Flaxman & Frieze 2004]. It means, there exists a constant C_r such that the diameter $\leq C_r \times \log n$ with high probability when n tends to ∞ (see Section 2.2.5.3).

2.2.4.3 Clustering Coefficient

In graph theory, a clustering coefficient (*Clustering Coefficient (CC)*) is a measure of the degree to which nodes in a graph tend to cluster together. This measure can be calculated for each vertex (local) and for the whole digraph (global). The local clustering coefficient of a vertex i is defined as the fraction of all neighbors of i that are also neighbors among themselves(triangles). It is obvious that if a vertex has less than two neighbors, its local coefficient is zero. The local coefficient clustering of a node i is given by: $C_i = \frac{\text{abs}\{(j,l)\in\overrightarrow{E}(\mathbb{D}^{\gamma}_{n,k,r\text{-}out}) : j,l\in N_D(i)\}}{deg_D(i)(deg_D(i)-1)}$. The global clustering coefficient is the average of all local clustering coefficient of each participating vertex is given by: $CC = \frac{1}{n}\sum_{i=1}^{n} C_i$. As the value of CC alone is not sufficient to determine that the digraph is highly or lowly clustered, a comparison with oriented edge density metric (ρ) is necessary. If CC is much higher than ρ, the digraph is considered highly clustered. Otherwise, the digraph is considered lowly clustered. The directed edge density is calculated by using the ratio between current number of directed edges and the total number of all possible edges. For approximate and exact $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraphs, the oriented edge density is given by $\rho = \frac{r\times n}{(n-1)(n-k)}$.

2.2.4.4 Other Properties

Following are some other properties which are important in improving information dissemination and optimization of blockchain P2P network:

- *Graph Generation Time*: It is the amount of time taken to generate a map of the given P2P network graph. This time is important when centralized protocols for topology construction are used. For distributed protocols, this property is irrelevant.

- *Maximum Inbound Connections*: It is the maximum inbound connections any node within the given P2P network has i.e., the node having the highest individual clustering coefficient. This property closely follows the clustering coefficient i.e., for higher clustering coefficient value, one can roughly expect a higher value of maximum inbound connection for a node within the given blockchain P2P network.

2.2.5 Simulation & Results

In this Section, we first explain our simulation techniques and then summarize our findings in brief for all properties. Armed with these simulated trends, one can model any given blockchain P2P network with optimal parameter values.

2.2.5.1 Energy Consumption Analysis

Synopsis: Our proposition of centralized topology control consumes less energy than the prevalent distributed topology control. Our solution is more adapted for energy constrained peers.

Methodology: We estimate the average energy consumption required for topology control via averaging total no. of topology control packets exchanged per node in the time frame of 24 hours. We calculate this energy consumption for peers using bitcoin blockchain as a reference example for distributed topology control. Further, we compare it with energy consumption of our proposed centralized topology control method under similar settings.

In general, the total number of messages in a network of n nodes, each having r outbound connections, in Δt hours, can be given as (*number of messages/hour/node/connection*)$\times n \times r \times \Delta t$.

For the calculation of bitcoin topology control energy consumption, we analyze all the cases when a given bitcoin peer transmit/receive control message(s):

1. When a node (new or old) establishes a connection with a remote node, it transmits its address with *addr* message.

2. Each node (new or old) broadcast every 24 hours its own address with an *addr* message to all connected nodes.

3. When node (new or old) receives an *addr* message (solicited/unsolicited), it forwards it (under conditions that entries in its address database are < 10) to two neighbors as an unsolicited *addr* message.

4. When a node (new or old) establishes a connection with a remote node, it can transmit a *getaddr* message.

When mapping bitcoin topology [Deshpande *et al.* 2018], it is observed that there are 10000 nodes approximately with 1% of it (100 nodes) being replaced every hour. Considering the P2P network topology (see Section 2.2.3), bitcoin follows an r-*out* topology with $r = 8$ (see Table 2.1). Hence, for bitcoin, the total number of transmitted topology control messages:

- Case 1: No. of new nodes/reconnecting nodes every 24 hours*8 (24*100*8).
- Case 2: Total nodes*8 per day (10000*8).
- Case 3: ((Total nodes-1)*2)*(Case 1 + Case 2).
- Case 4: Assuming new/reconnecting nodes send at least 1 *getaddr* and get 1 *addr* response every 24 hours to complete their database (24*100*8*2).

Total messages/24 hours = Case 1 + Case 2 + Case 3 + Case 4
Average no. of messages per peer per day = Total messages per day / total peers.

Next, we also analyze all the cases for our proposed solution, when a peer transmit/receive control message(s) (vertically), for a similar network setting and size as that of bitcoin network and for same number of $r = 8$ outbound connections with 1% of nodes being replaced every 1 hour:

1. When a new node arrives in the network, it transmits to the server(s) *get_ neighbors* message. Then, it will receive a set of *neighbors* message. Since 1% nodes are replaced every hour in the bitcoin network [Deshpande *et al.* 2018], we assume the same number for our proposed solution.

2. - All nodes in the blockchain P2P network transmit *keep_ alive* message periodically to certify their liveliness. The *keep_ alive* message is for server(s) to know the nodes' liveliness. If nodes are not alive, they stop to send the *keep_ alive* message and thus, the server can disconnect them. This way, the server(s) can keep the r-*out* graph up-to-date.

- If the periodicity of the *keep_ alive* message is very short, nodes will consume more energy. However, the servers can detect the offline nodes quicker. If the time periodicity is very large, we save more energy but the network is not up-to-update. Thus, there is a trade-off between energy consumption and the up-to-date server vision to maintain the r-*out* graph.
- To configure the time periodicity of the *keep_ alive* message, the value depends on the type of nodes, the available energy, the expected performance profile and the application type. For e.g., for low energy IoT devices, due to limited energy availability, it is not recommended to have a low time periodicity (frequent transmits). For other types of nodes like higher energy IoT devices or mobile devices, energy availability is more, hence the time periodicity can be low/reduced (more frequent messages can be transmitted in a given time-frame).
- Further, when a node is offline, its neighbors can detect this event when attempting data exchange with it as no *ack* message is received. Thus, the number of outbound connections for the neighbors is reduced (below the threshold r). Subsequently, they would inform to the server to select other neighbor(s). When this approach is combined with *keep_ alive* message technique, we further reduce the time period of the window when the exact knowledge of the P2P network structure is not available.

  Indeed, during this time window, the r-*out* graph may become approximate (see Section 2.2.3.4) or worse if number of nodes going offline increase. Consequently, this may directly impact the properties of the original P2P network in terms of connectivity, diameter, reconfigurability, etc. If this time window is increased, the worst case time period of network not being r-*out* is also increased.

  Therefore, to reduce this time period without much overhead and for simulation purposes, we set the time periodicity of *keep_ alive* message as 30 minutes. Of course this value is configurable and can be fixed according to the application run on the blockchain. In fact, real-time applications may require real-time up-to-data topology, hence short period can be used i.e., frequent *keep_ alive* messages.

3. If a peer should be connected to r-neighbors and the current outbound connections are less than the critical threshold, then it will transmit *get_ neighbors* request message again to the server(s). Consequently, it will receive a set of *neighbors* message from the server(s). Since this

number is difficult to estimate and since 1% of network nodes are replaced every hour [Deshpande *et al.* 2018] in bitcoin network, we assume the number of nodes to have less than r outbound connections to be $1\% * r$ assuming an r-*regular* digraph i.e., each node has equal number of inbound-outbound connections ($r_{out} = r_{in}$).

4. Every peer transmits once per day to the server the status of its outbound connections. The objective of this task is to update r-*out* digraph once per day.

The total number of transmitted control messages using our solution are:

- Case 1: No. of new nodes/reconnecting nodes every 24 hours (24*100*2).
- Case 2: Total nodes*2*24 messages per day (24*10000*2).
- Case 3: Total nodes*probability to loose at least 1 connection per hour*24*r*2 (24*10000*1%*8*2).
- Case 4: Total nodes (10000).

Average no. of messages in our solution per peer per day = Total vertical messages per day (Case 1 + Case 2 + Case 3 + Case 4)/ total peers.

For both cases, energy consumption per peer per day due to the control topology management = average no. of messages*energy consumption per message. As the total energy consumption per peer is directly proportional to the total number of control topology messages exchanged (assuming energy consumption per message is nearly constant for empirical analysis), the energy efficiency of our proposed centralized solution compared to distributed approach can be easily affirmed through the simulation results presented in Figure 2.4.

In Figure 2.4, the x-axis represents the two approaches of topology control i.e. Distributed (Bitcoin) and Centralized. The y-axis is *log-scaled* and represents the total control messages sent per 24 hours per node for both distributed and centralized approaches. From Figure 2.4, the centralized solution for topology control is $\approx 3.7k$ times energy efficient compared to the distributed approach. The overall efficiency of the centralized solution slightly reduces when power consumption of upper tier nodes (see Section 2.2.2.1) is also considered. For both cases, the average throughput per peer per day for topology control management = average no. of messages per peer per day * average size of topology control message.

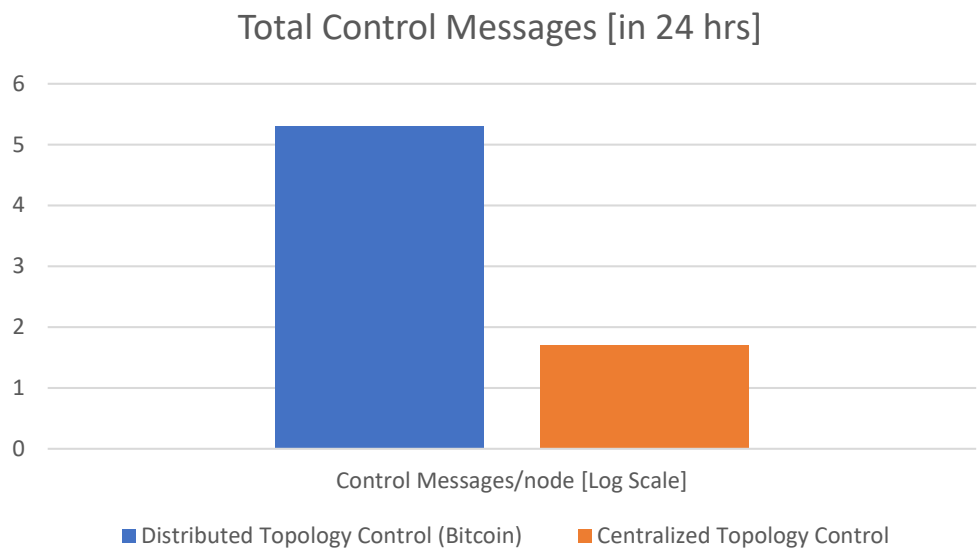


Figure 2.4: Control Messages per 24 hours per node, Distributed vs Centralized.

2.2.5.2 Other Simulation Parameter and Synopsis

The results presented in this Section are a part of a larger group of simulation results. Therefore it is imperative to summarize our findings. For our *r-out* digraph generation techniques, we experimented with following types of models:

- Approximate *r-out* digraphs: Erdős-Rényi digraphs (Binomial)
- Exact *r-out* digraphs: Preferential Attachment digraphs ($\alpha = 1, 10, 20$), Binomial Distribution digraphs

The digraphs generated were evaluated under 5 broad parameters, viz., Diameter, Connectivity, Graph Generation Time, Degree of Clustering, Maximum Inbound Connections. These parameters helped us to find the best suitable algorithm for optimally modeled blockchain P2P network.

For simulation, each type of sample digraph was generated varying the number of participating nodes n, the number of non-contributors k (%)and r outbound connections per node. This sampling was repeated 100 times, varying number of nodes n from 1000 to 10000 (step size = 1000). The percentage of wallets k was varied from 0 to 75% (step size = 25%). The outbound connections were varied with $r = 2, 4, 6$. The tools used to accomplish these simulations were Python 3.2 with NetworkX Library 2.1.

For easy understanding of each digraph's performance across different parameters, Table 2.2 illustrates the scores for each individual digraph type

Table 2.2: Comparison of different types of digraphs, PA = Preferential Attachment.

Digraph	Connectivity	Clusters*	Max IC*	Gen. Time*
Erdős-Rényi	3	3	1	4
PA ($\alpha = 1$)	5	2	2	1
PA ($\alpha = 10$)	5	5	3	1
PA ($\alpha = 20$)	5	5	4	1
Binomial Exact	5	5	5	5

under each parameter. The higher the score, the more suitable the digraph type is. (** indicates a higher score for lower entity value. Diameter is not compared as relative variation was statistically insignificant.*)

From Table 2.2, we can conclude that, within our simulation range, Binomial Exact r-*out* digraph excels across all categories with the highest score followed by Preferential Attachment with high α value. From our simulations, we also found that at higher α value, the Preferential Attachment Exact r-*out* digraph has properties similar to Binomial Exact r-*out* digraph.

2.2.5.3 Diameter

The variation of the diameter was observed varying the values of total nodes n, the percentage of non-contributors k and the outbound connections (r, above the critical value r_c). Synopsis of the simulated trends is presented in the graph below. To summarize it was found that when increasing the number of participating nodes n in the network, the diameter increased till a maximum threshold C_r, as indicated in Section 2.2.4.2. Contrarily, increasing the outbound connections r reduced the diameter. Increasing the number of non-contributors k had a similar effect on reducing the diameter, however, it was more drastic.

In Figure 2.5, the percentage of non-contributors and number of outbound connections are varied for different network size. We can see that for the same percentage of non-contributors k across various network sizes, the diameter reduces by at least 1 when outbound connections r are increased by 2. This is because average connection probability increases when r is increased thereby making the digraph more densely connected. Further, for the same number of outbound connections r across various network sizes n, the diameter decreases at least by 2 when the percentage of non-contributors k is increased from 25% to 75%. This is because of the clustering effect (contributors at the center and non-contributors at the periphery), as a result of more non-contributors.

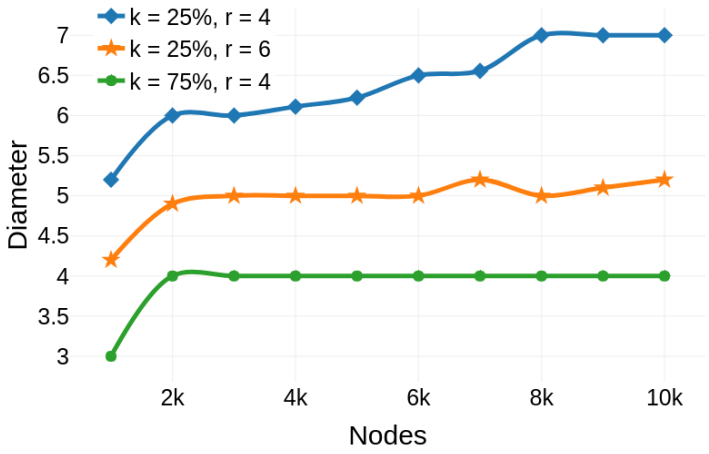


Figure 2.5: Diameter Variation.

Each individual plot saturated at a maximum upper bound value.

2.2.5.4 Network Connectivity

With reference to Section 2.2.3.4, the network connectivity was tested for both types of *r-out* digraphs, i.e., Approximate and Exact. For the Exact *r-out* digraphs, we obtained a connected digraph when $r \geq 2$. For the Approximate *r-out* digraphs, when $r > r_c$ (from equation 2.10), we obtained a connected digraph with a very high probability ($\geq 90\%$).

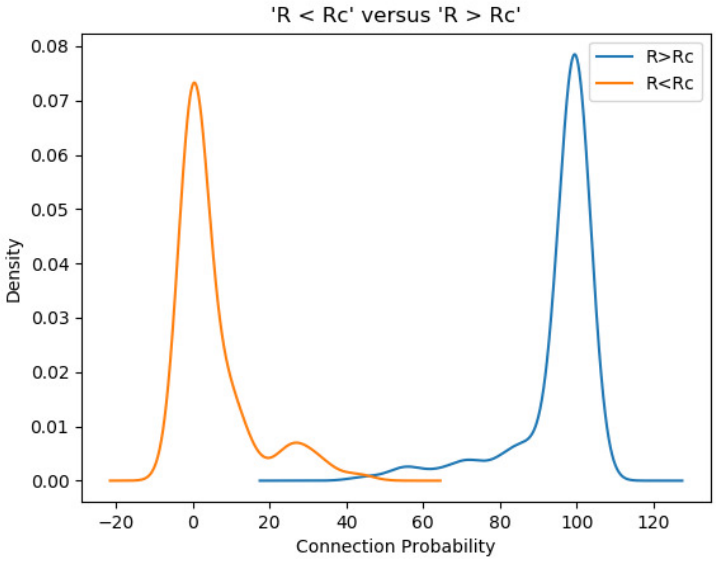
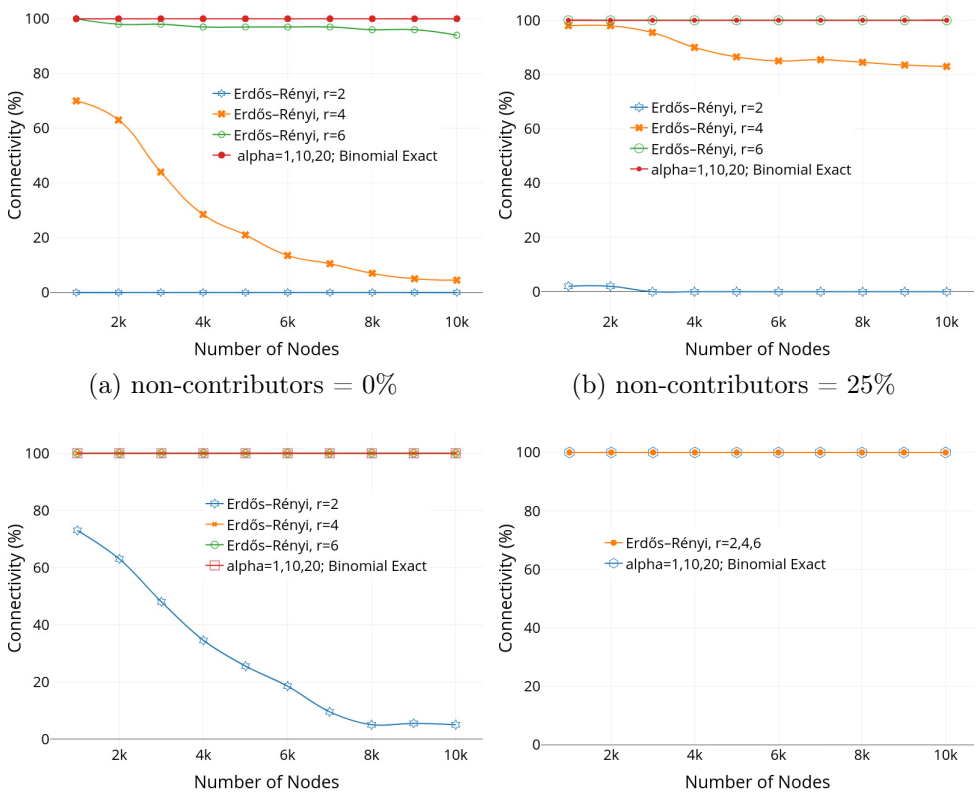


Figure 2.6: Connectivity histogram for approximate *r-out* digraphs.



(a) non-contributors = 0%

(b) non-contributors = 25%

(c) non-contributors = 50%

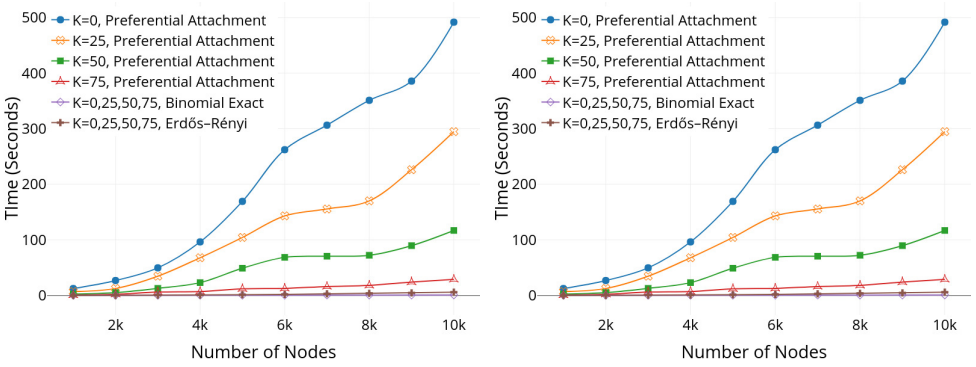
(d) non-contributors = 75%

Figure 2.7: Connectivity of *r-out* digraphs.

Figure 2.6 depicts the histogram of Approximate *r-out* digraphs for $r > r_c$ and $r < r_c$. There are two peaks visible showing that most of digraphs are disconnected when $r < r_c$ and most of the digraphs generated were connected when $r > r_c$. Figure 2.7 shows the relative connectivity for the Approximate and Exact *r-out* digraph. The values of r tested were $2, 4$ and 6. Binomial and Preferential Attachment digraphs ($\alpha = 1, 10, 20$) were simulated for Exact *r-out* digraphs and they were connected across all the simulations. The Approximate *r-out* digraphs (Erdős-Rényi) were connected with more probability when r value increased or when the number of wallets (non-contributors k) increased (attesting the value of r_c from equation 2.10 of Section 2.2.4.1.2).

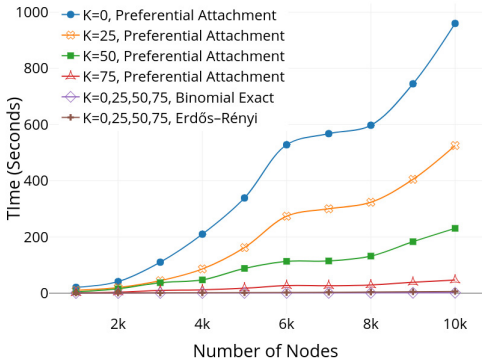
2.2.5.5 Graph Generation Time

The figure 2.8 represents the time in seconds required to generate digraphs. It is evident that time for graph generation increases with increase in the number of nodes n or number of outbound connections r as both causes increase in total edges to be formed in the P2P network. Increasing the number of non-contributors k reduces the graph generation time. The graph generation time



(a) outbound connections = 2

(b) outbound connections = 4



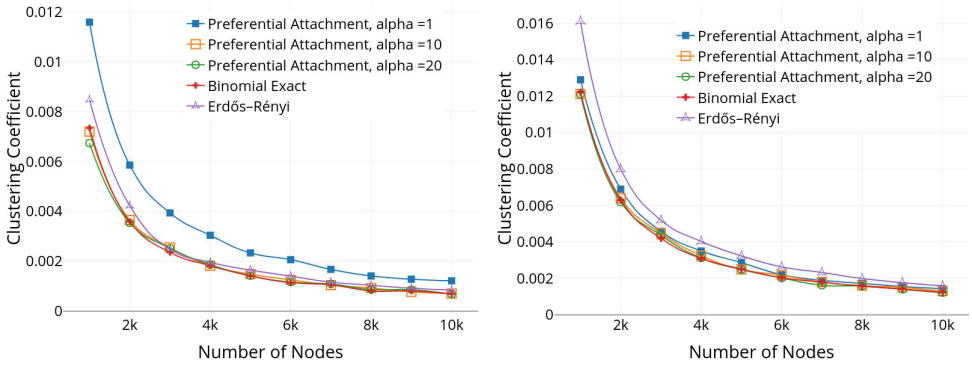
(c) outbound connections = 6

Figure 2.8: Graph generation time (K expressed in %).

for Approximate r-*out* graph (Erdős-Rényi) was less compared to Exact r-*out* graph (Preferential Attachment). This is obvious given the fact that there are additional constraints to make the digraph Exact r-*out*. However, this time can largely be reduced by avoiding vertices whose in/out degrees are saturated w.r.t threshold. Binomial Exact r-*out* digraph's reduced generation time illustrates this (see Section 2.2.3.4.4). We can also observe that the change in the number of non-contributors k affects Preferential Attachment digraph only.

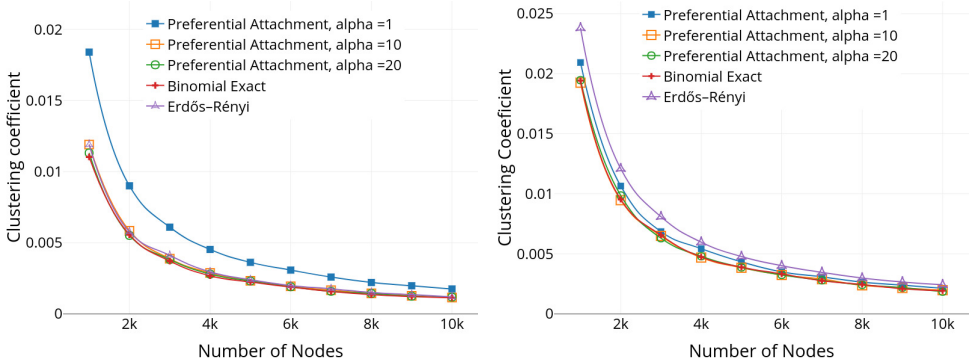
2.2.5.6 Clustering Coefficient

Figure 2.9 shows the plots for clustering coefficient. The trends depicted are for outbound connections $r = 4, 6$ and for non-contributors $k = 0\%, 50\%$. The coefficient value *decreases* with an increase in the number of total participating nodes n while it *increases* when outbound connections r or non-contributors k are increased(more clustering). Preferential Attachment with lower α value has most clusters as the initial weight assigned to each node is low during



(a) outbound connections = 4, non-contributors= 0%

(b) outbound connections = 4, non-contributors= 50%



(c) outbound connections = 6, non-contributors= 0%

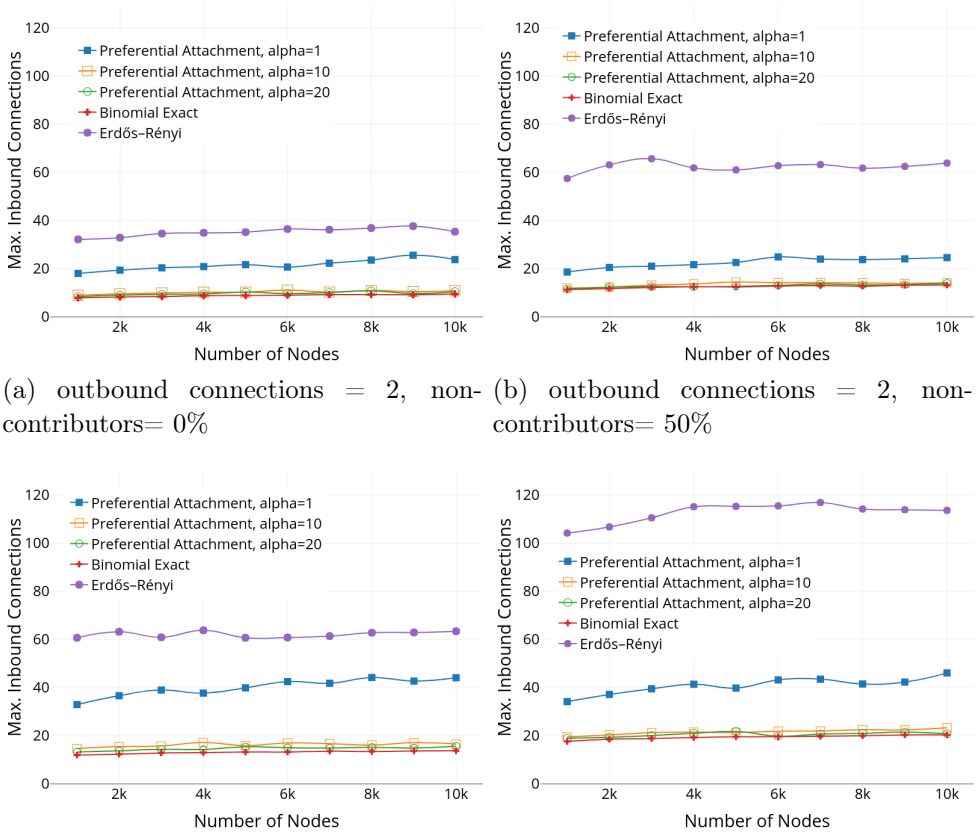
(d) outbound connections = 6, non-contributors= 50%

Figure 2.9: Clustering Coefficient.

the graph generation (see Section 2.2.3.4.5). However, Approximate *r-out* digraph (Erdős-Rényi) tends to cluster more when non-contributors k in a given blockchain P2P network increase. The Exact *r-out* digraph (Binomial) performs better in this case as well with a low value of the clustering coefficient.

2.2.5.7 Maximum Inbound Connections

Figure 2.10 illustrates the plot of maximum inbound connections (maximum clustering) a node can have within the blockchain P2P network (when γ is not fixed). Here, in this case, Approximate *r-out* digraph (Erdős-Rényi) performs far worse compared to Exact *r-out* digraph. Maximum inbound connections are fairly constant and do not vary to a great extent when the number of participating nodes n is increased. However, this value *increases* when the number of non-contributors k or the outbound connections r is increased. For example, for Erdős-Rényi, this value nearly increases by 100% when non-



(a) outbound connections = 2, non-contributors= 0%

(b) outbound connections = 2, non-contributors= 50%

(c) outbound connections = 4, non-contributors= 0%

(d) outbound connections = 4, non-contributors= 50%

Figure 2.10: Max inbound connections.

contributors k increase by 50% (for outbound connections $r = 4$).

2.3 Blockchain Framework Evaluation

Blockchain framework evaluation is an important step for two reasons. Firstly, to affirm the findings of the blockchain framework modeling, and secondly, to assess an existing framework for its usability in a given use-case(s). For evaluation of any blockchain framework, it is imperative to analyze all performance metrics of a given blockchain, with underlying P2P network being the most important. Most, if not all, performance metrics in a blockchain are directly related to the performance of underlying P2P network. Therefore, it is critical to evaluate the underlying P2P network. The blockchain performance is directly impacted by the P2P network properties such as network diameter, average number of connections per node, connectivity, clustering.

In the context of blockchain, underlying P2P network can be visualized as graph where nodes represent vertices, and connections between them are represented as edges. Further, for evaluation of various properties, mapping of P2P network topology is important as a snapshot of topology can easily be utilized for calculating the diameter, cluster size, connectivity. It is important to note that in blockchain, the topology discovery is deliberately disabled for security reasons but for our perspective, we just need an approximate snapshot which has identical network properties for evaluation of various performance parameters.

Hence, to map topology of a given blockchain framework, we use its respective neighbor discovery protocol. In Section 2.3.1, we elaborate our strategy of implementation on non-permissioned blockchain framework of bitcoin.

2.3.1 BTCmap Framework

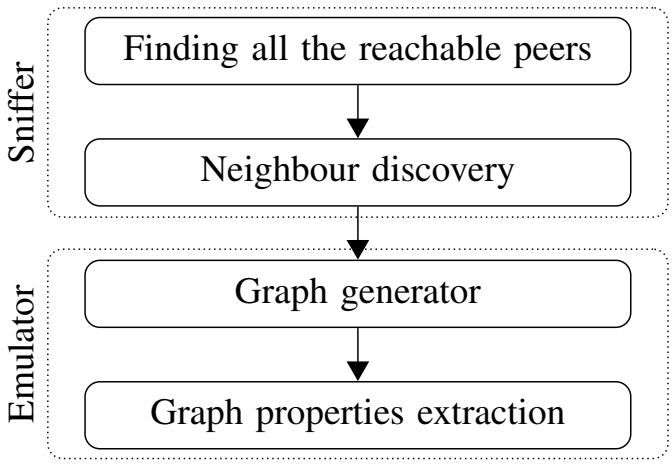


Figure 2.11: BTCmap framework.

Our BTCmap framework is organized into two modules viz., the sniffer module and the emulator module as illustrated in Figure 2.11. The sniffer module interacts with real Bitcoin network and hence has timing constraints as network topology is constantly changing. This module is further divided into two submodules viz., submodule to find all the reachable peers and submodule for neighbor discovery. The second module i.e, emulator module works on output from the sniffer module and is also further divided into two submodules viz., submodule for graph generation and submodule for extraction of the properties of the graph. In the following sections, we explain each submodule.

2.3.1.1 Finding all the reachable peers

To find current online reachable peers in the Bitcoin network, we use Bitnodes' server [Yeow & Lopp 2013]. This server is dedicated to estimate the size of the Bitcoin network by sending *getaddr()* request messages recursively to find all the reachable peers, starting from a set of DNS seeds. As Bitnodes server has live information about all the reachable peers, BTCmap requests the same via Representational State Transfer (RESTful) Application Programming Interface (API)[1]. Bitcoin mapper (BTCmap) sends a simple GET request to Bitnodes' server and it gets a response which includes the list of reachable active peers. In this work, we considered only peers with IPv4 addresses. We skipped Onion and IPv6 peers as we lacked proper network infrastructure to test and integrate them. These skipped peers accounted for 15-18% of total reachable peers (see Table 2.3).

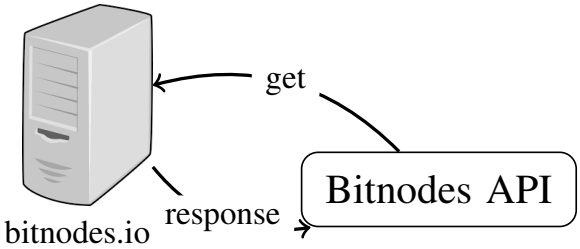


Figure 2.12: Getting active peer list using Bitnodes

Each list of reachable peers given by Bitnodes remains valid approximately for 5 minutes and 40 seconds. After that, a new list becomes available. However, comparing the previous 10 snapshots, we found that less than 1% of peers change (see Section 2.3.2.1). This effectively increases the fidelity of the given list and hence the time limit to process it.

2.3.1.2 Neighbor Discovery

The objective of this submodule is to build local address databases of all peers. Once BTCmap has the list of reachable active peers, it recursively sends *getaddr()* request messages to everyone. As a result, the peers respond with multiple *addr()* response messages. Each *addr()* response message contains up to 1000 peers. As for each peer, the size of the local address database is 81920 entries (see [Deshpande *et al.* 2018]) and each *addr()* response is different (randomly selected from different buckets), BTCmap sends multiple

[1] method of communication between client and server entities

getaddr() request messages to retrieve all possible entries for given peer. After making a request, *addr()* response message was received within a specified timeout period. Some peers can reject BTCmap connection requests due to saturation of their inbound connections quota. For those peers, BTCmap waits to get a free slot for inbound connection and then retries with a subsequent connection request. BTCmap continuously aggregates received *addr()* messages from each peer, verifies that advertised peers in the *addr()* messages are still online, and concurrently remove duplicates. This results in the recreation of the local database of possible neighbors for each peer.

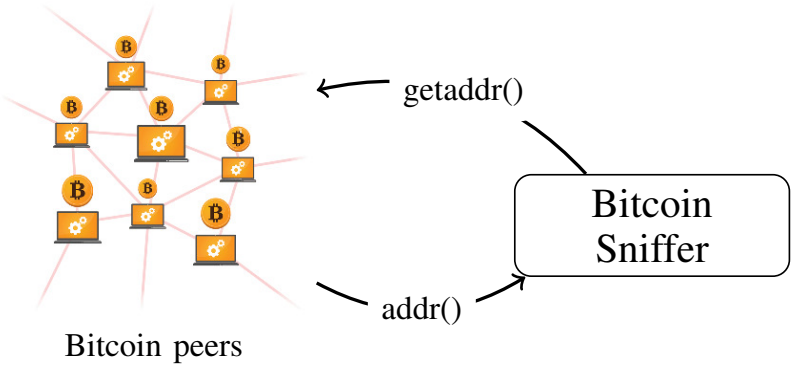


Figure 2.13: Neighbor Discovery

2.3.1.3 Graph Generator

This submodule works on the output from neighbor discovery submodule. After having a list of probable neighbors (local addresses database) for each online peer, they are randomly classified into tried and new buckets following Algorithms 1 and 2. The next step consists of selecting outbound neighbors for every peer based on Algorithm 3. Consequently, the selected outbound neighbors for each peer are then recorded in a large adjacency matrix.

The adjacency matrix represents a possible snapshot of real Bitcoin network. The real snapshot may differ from the one generated by BTCmap because the random generator used by BTCmap doesn't use the same seed value as used by the peers. Indeed, BTCmap can generate many possible yet different snapshots varying seed values and then compute the average snapshot. As the real network topology and simulated/generated by BTCmap use the same input data and algorithm model, the network properties like connectivity, diameter, cycles, etc., are close to each other.

2.3.1.4 Graph Properties Extraction

This submodule is the last process in BTCmap yet most important. This submodule works on the adjacency matrix obtained from Graph Generator. The submodule extracts many important network properties such as the connectivity of the network, the diameter of the network, the average probability of connection, etc. These properties help to determine the degree of unevenness or clustering in a real Bitcoin network. Full and detailed implementation of this submodule for a more thorough study of Bitcoin topology remains the prime objective for our future work.

2.3.2 Experimental Results

We have implemented our BTCmap in Python environment [Documentation 2000] ensuring compatibility across Bitcoin Core versions. For our experiments, we have only worked on Bitcoin peers with IPv4 addresses. In this Section, we show and discuss the performance of each submodule of BTCmap.

2.3.2.1 Reachable Active Peer Detection Results

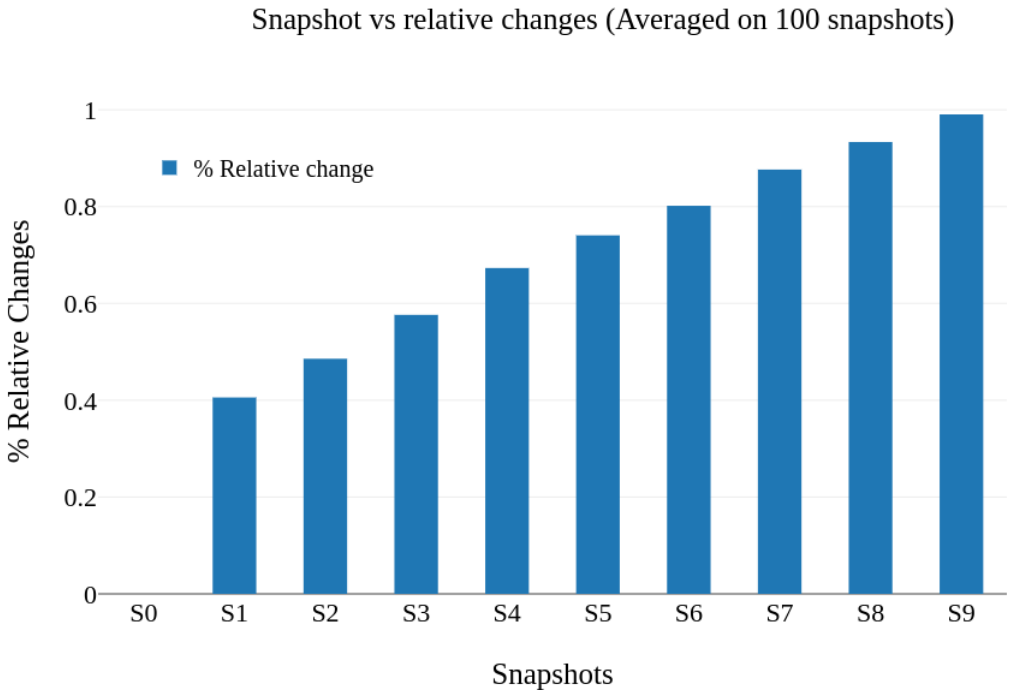


Figure 2.14: Relative changes with S0 as reference list.

The objective of experiments of this part is to show the fidelity of the peer-list retrieved from Bitnodes' server. As Bitnodes updates the reachable active Bitcoin peer list every 5 minutes 40 seconds, we had first recorded 10 consecutive peer lists for a total duration of 56 minutes 40 seconds and for each peer list, we measured the difference compared to the first obtained list (Reference list - S0). Then, We repeated this total process 100 times. The averaged results are depicted in Figure 2.14. We can remark that each new list does not vary significantly. This is evident from the difference between two consecutive lists which is less than 0.4% in all cases. As we go further, difference increases. For instance, the difference between reference list (S0) and list which was issued 10 minutes 20 seconds (S2) after is approximately 0.5%. We also observe that changes are less than 1 % even when 10 consecutive peer-lists were considered. Therefore, to remain within 99% confidence interval, real-time neighbor discovery can only last for a duration less than or equal to *(5 minutes 40 seconds)*10*.

The next experiments show the percentage composition of IPv4, IPv6, Onion peers in a Bitcoin network extracted by BTCmap from Bitnodes' server. The Table 2.3 depicts the statistics of first 6 months for the year 2018. We can see that the majority of the peers use IPv4 addresses (more than 83%).

Table 2.3: Percentage of IPv4, IPv6, Onion peers (H1 2018).

	Jan.	Feb.	Mar.	Apr.	May	June
IPv4 %	83.52	83.59	85.01	83.47	83.65	83.76
IPv6 %	13.98	13.89	12.58	13.64	13.5	13.27
Onion %	2.5	2.52	2.41	2.89	2.85	2.97

2.3.2.2 Neighbor Discovery Results

In this part, for the first set of experiments, we measure the necessary time duration for recreating local address databases of all active reachable peers. For this purpose, once the peer-list is retrieved from Bitnodes' server, BTCmap initiates connections to all peers and send recursively multiple *getaddr()* request messages in a window of 200 seconds which is the upper bound to receive a response. Those peers who reject the BTCmap's connection request, are queried again in subsequent iterations. The cumulative number of responding peers for each iteration is shown in Figure 2.15. We see, from the plot that after 160 seconds into any iteration, the probability to receive a response from a new peer is statistically negligible. Therefore, We fixed our iteration duration to 200 seconds, i.e. BTCmap waits for 200 seconds for each peer after

sending the request messages. We observe that till 6^{th} iteration, the absolute count of peers responding is significant. However, from 7^{th} iteration to the 17^{th} iteration, it reduces greatly.

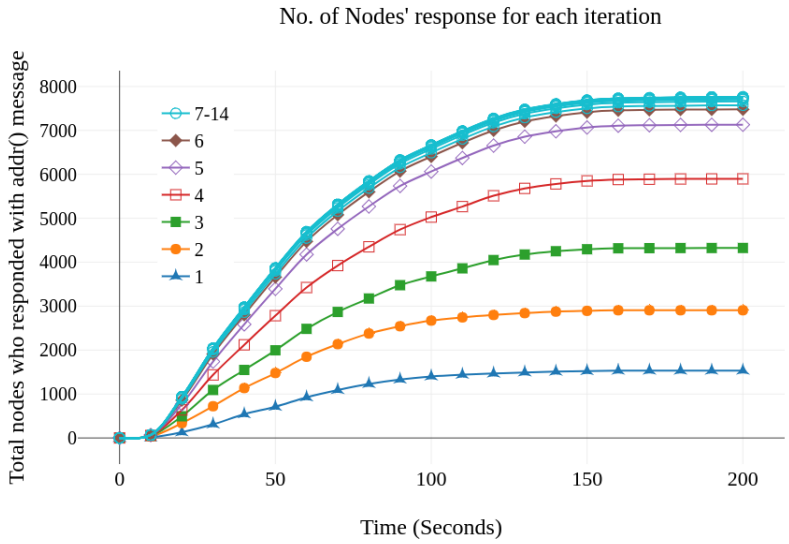


Figure 2.15: Cumulative peer's response with time - *For 17 iterations.*

The reduction in peers' responses is also demonstrated in Figure 2.16. The Figure represents the progress for a typical mapping of Bitcoin topology. The graph slope is quite steep for first 6 iterations. The cumulative response passes the 90% mark before the end of the 6^{th} iteration for the queried peers. After that, the 95% cumulative response mark is passed only at the end of the 1% change window, i.e., in the 17^{th} iteration.

Because of the flooding of *getaddr()* request messages, peers (according to their processing capabilities and network delays) respond with varying number of *addr()* response messages. These messages contain the peers which the target peer knows about. BTCmap clubs together all these response messages for each peer, removes duplicates and inactive addresses to give us the local recreated database of active IP addresses (probable neighbors) for each peer. Figure 2.17 shows the plot of the number of active IP addresses in recreated databases versus the number of peers.

2.3.2.3 Graph Generation Results

By applying our emulator on the output collected data from the sniffer module, an adjacency matrix is generated. Figure 2.18 shows a snapshot of Bitcoin network topology generated on 14th June 2018 at 15:53 CEST. In the Figure, peers are represented by red vertices and outbound/inbound connections by

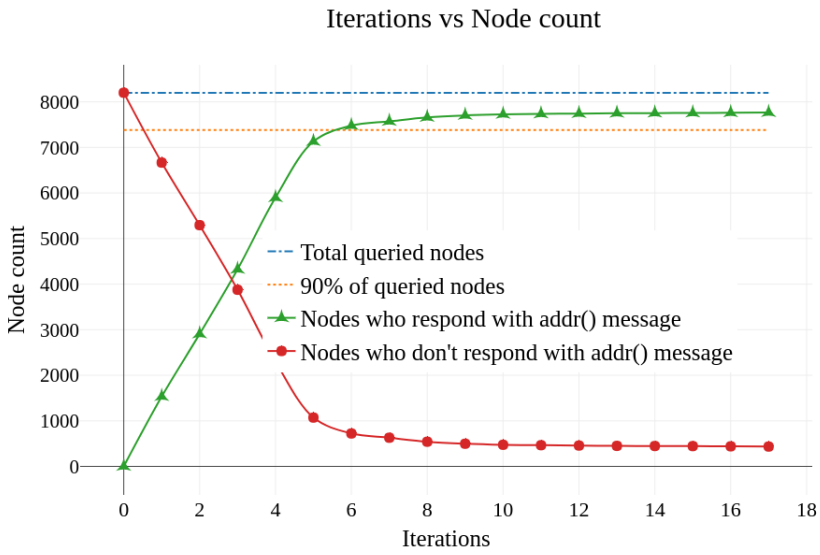


Figure 2.16: Number of peers responded against iterations.

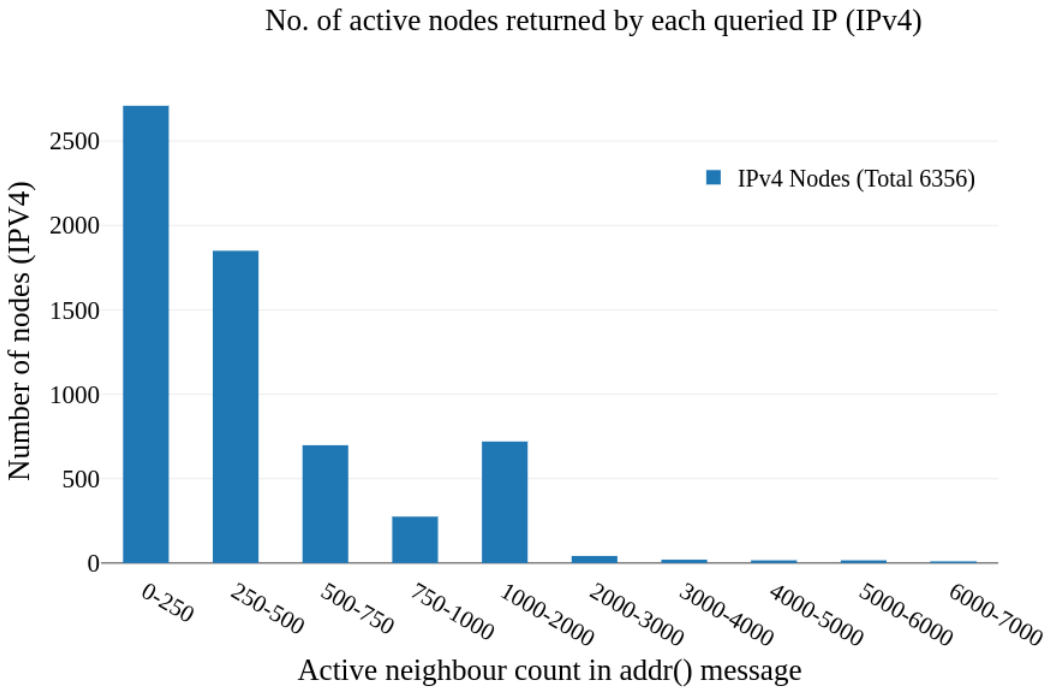


Figure 2.17: Active IPs in *addr()* message vs no. of peers returning this count.

edges. We can see that the topology is very dense. By running a simple algorithm to visit every peer using Breadth First Search [Belova & Ouyang 2017] on the snapshot, we have found that the snapshot was connected with a di-

ameter value of 7. When compared to our asymptotical findings from our blockchain modeling framework (see Section 2.2.3), the connectivity as well as the approximate diameter results match. Thus, attesting our blockchain modeling framework as well.

In a nutshell, the current outbound/inbound limits for bitcoin network are sufficient to ensure connectivity for this snapshot. BTCmap cannot be compared to any existing modeling tool as [Miller *et al.* 2015] do not support protocol version of Bitcoin higher than (0.10) and [Biryukov *et al.* 2014] has fundamental architecture issues. It is therefore not possible to elaborate further on efficiency and accuracy of BTCmap.

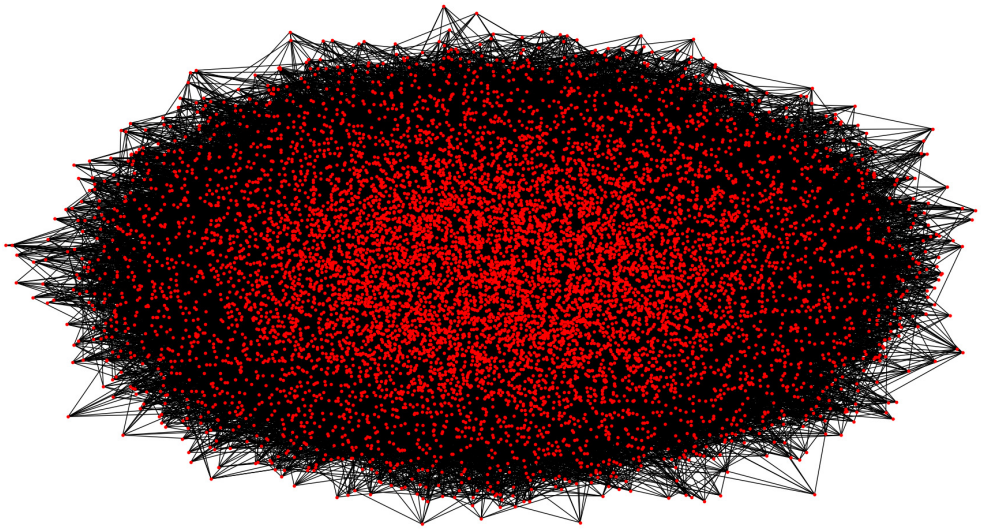


Figure 2.18: A visual snapshot of Probable Bitcoin Topology on 14 June 2018, 15:53 CEST .

Summary: In this chapter, we explained how blockchain can be used as distributed database for securing *transactions*. We also addressed the main problem of blockchain adoption i.e., modeling of new blockchain framework while evaluation of already available blockchain frameworks. However, to holistically secure a *transaction*, additional safeguards are necessary. We explain these in the next chapter.

Chapter 3

A Blockchain+SE Combinational Approach to Secure Transactions

Contents

3.1 Introduction to the Combinational Approach

The classical ACID properties are important to carry out any reliable *transaction*. However, in the modern implementational contexts, many more important properties are to be fulfilled for a fool-proof execution of a *transaction*. In the context of holistic transaction security, blockchain solves many related

problems. However, given its high overhead and low penetrability, we propose the use of Secure Element (SE) (see Section 1.5).

SE in this context can be used to reliably satisfy the properties of security, authenticity, TEE, TSE. It can be used as *root of trust* as it can be connected directly at the lowest edge of any given system thereby securing data at root. Even though other technologies exist to satisfy these properties, the compelling reason to adopt SE is its *Secure by Design* approach where the security focus is interwoven in its bare hardware design rather than as an add-on or software-based approach.

In this chapter, we combine the technology of blockchain and SE. In the next three sections, we apply this combinational approach to 3 distinct fields i.e., Smart Grids, Smart Vehicles, Internet of Things.

3.2 Combinational Approach applied to Smart Grids

Synopsis: In the context of smart grids, Demand Response (DR) is used to manage energy imbalance by smoothing consumption peaks through voluntary rationing of energy by participants. However, it largely remains centralized and opaque with little to no traceability. To resolve this, we apply our blockchain-SE based combinational approach to propose a novel framework [Deshpande *et al.* 2020] in which a consortium of DR allotters and certifying authorities maintain the blockchain. This brings in more transparency, traceability, and complete decentralization along with trustlessness, non-repudiation, and immutability. Further, the framework uses distinct components/concepts like Escrow Accounts, Applied Smart Contracts in unison to fix the impediments of previous blockchain-based propositions. Next, we propose a fair and efficient DR allotment mechanism for a distributed DR marketplace whose execution time is less than 1 minute for more than 20,000 participants. Further, through simulations, we show the impact of different parameters on it and demonstrate its ability to delicately balance various paradigms of DR metrics. We also present our findings on system reliability and its inordinate effects on DR allotment metrics.

3.2.1 DR Framework and its Components

In context of smart grids, A typical DR session involves a wide range of heterogeneous systems coordinating together. Our proposed framework illustrated in Figure 3.1 likewise uses a wide range of components and concepts which are

described in detail in this Section through which it achieves complete decentralization of the DR session viz., allotment, monitoring, settlement (explained in Section 3.2.3). In our framework, Transmission System Operator (TSO)s and market regulators (tier 1) maintain a blockchain which serves as a backbone database for the DR participants (tier 2). The participating bids for DR are collected by all the tier 1 participants and allotment is collectively decided by them using consensus. To have a synchronized gate open/close timing, each tier 1 participant is equipped with a Global Positioning System (GPS) clock. For *proof of origin*, tier 2 participants are equipped with an SE.

3.2.1.1 Blockchain

In our framework, we propose to use a private permissioned blockchain because, in the case of data transactions, the prerequisite access permissions are imperative to guarantee trusted results and privacy.

Next, blockchain platforms, in general, are optimized for a variety of use-cases based on different end-user needs. For e.g., some platforms (Hyperledger Fabric [Cachin 2016], Multichain [Greenspan 2015]) allows efficient data/finance transaction management while some platforms (Ethereum [Dannen 2017], Hyperledger Burrow [Monax]) are optimized for running code. In our work, we propose to use blockchain platforms optimized for efficient transaction management as there are more data/finance transactions to manage in a given DR session compared to only one code running requirement (DR allotment model) [Hasse *et al.* 2016] (see Section 3.2.1.2).

In our framework, blockchain serves as an all-in-one backbone. There are two tiers of participants, i.e., tier 1 participants maintain the blockchain and tier 2 participants access the Blockchain as a Service (BaaS). With this two-tier system, the issue of security vulnerability does not arise because the blockchain is maintained by tier 1 participants only (i.e. blockchain maintenance is not split across tiers). Next, this two-tier system helps to increase trustworthiness further as all the participants in tier 1 are trusted and only admitted after high scrutiny.

Tier 1 consists of TSOs and Certification Authority (CA)s who form a fully connected P2P network over a secure and reliable Virtual Private Network (VPN). CAs in tier 1 can either be DR marketplace regulators run by governments or delegated organizations acting as independent watchdogs to monitor DR marketplaces. Although the blockchain is accessible to all the participants of the DR marketplace, the writing rights (permission to form new blocks) is accorded only to tier 1 participants. Tier 2 consists of independent aggregators in the DR session who access blockchain as BaaS through tier 1 participants. During a DR session, participating aggregators (after monitor-

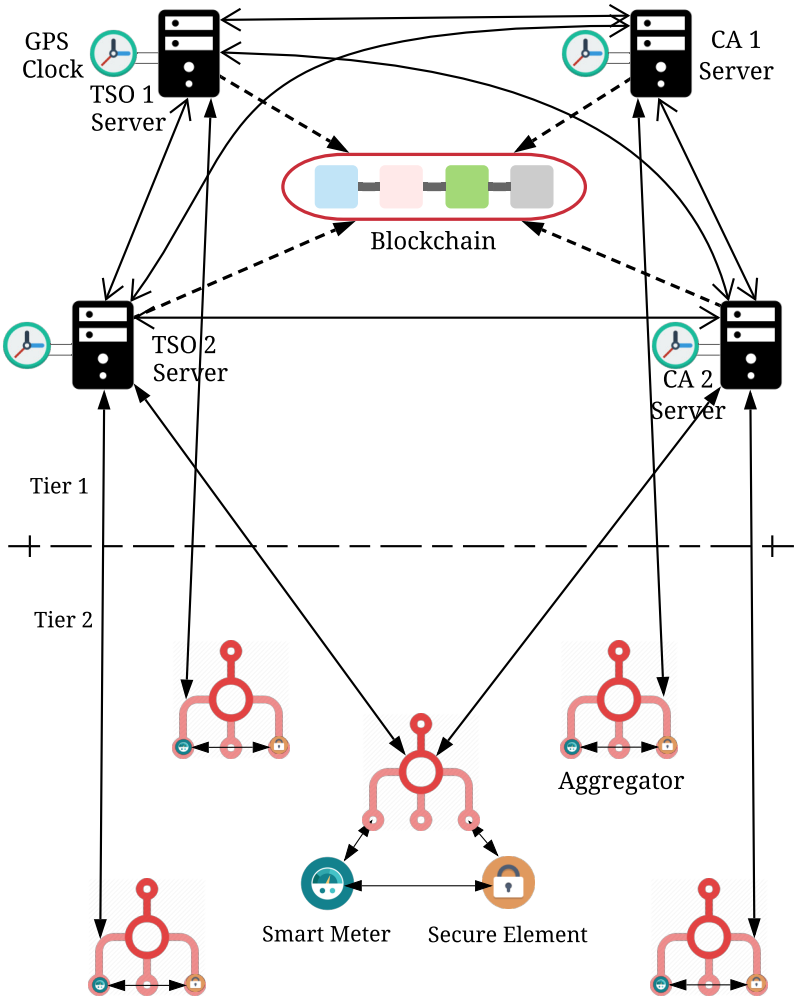


Figure 3.1: Blockchain-based framework for DR marketplace management.

ing the consumption pattern) pass their respective *calculated* bids to one or more tier 1 participants. However, to be resilient against tier 1 node failures, we insist that they always pass their bids to more than one tier 1 participants. Further, tier 1 participants maintain a synchronized transaction pool. All the bids received individually by tier 1 participants are pushed in their respective individual transaction pool. As all these transaction pools are synchronized, the changes appear in transaction pools of other tier 1 participants as well.

The transaction pools synchronization can be done using Reliable Multicast [Floyd *et al.* 1997]. Many different protocols are implementing Reliable Multicast [Diot *et al.* 1997]. A Reliable Multicast enforces a total order of the updates preserving the coherency of the distributed information. The rationale behind the proposition of common synchronized transaction pool in our

framework is to ensure coherent output during DR allotment as the selection model is applied by each tier 1 participant in a distributed setting on the same set of bids from the synchronized transaction pool.

Further, a model to be used for selecting the appropriate subset of aggregators during the DR session is first proposed by TSO. The model is then subsequently verified by the CAs in tier 1 through consensus and then added to the blockchain. Subsequent DR sessions then can use the newly certified model. The certification of the models helps to add an extra layer of security. For example, the certified model can favor aggregators based on given parameters (like reliability, consistency, punctuality) but the verification and certification of the models make sure that they are not biased or partial against invalid criteria (like favoring aggregators affiliated to political parties). As CAs collectively validate the model (through consensus), it increases the transparency between the aggregators and TSOs. Adding the certified selection model on the blockchain subsequently makes it immutable thereby securing future DR sessions.

The important property of blockchain which makes it indispensable in our framework is that it allows the DR participants to apply consensus for different decisions. The property of collective decisions is very unique to a blockchain. In the context of DR allotment, this collective decision property through consensus is of prime importance. This is because if a single entity (TSO) is the sole decision-maker, the decentralization of subsequent steps makes little sense. Even if decentralization solves trust issues for subsequent steps (e.g. payment), the prospect of not being allotted any DR quota at first place outweighs everything.

Further, consensus decisions in blockchain are inherently irreversible because of the collective nature of the decisions. Connecting this with the linking structure of blockchain, any data-editing possibility is ruled out till > 50% network deviates from its intended behavior. This guarantees immutability and a complete trace of past DR sessions. Next, features like native support for asset handling, multi-signature wallets (see Section 3.2.1.3) makes it apt in the context of DR.

3.2.1.2 Applied Smart Contracts

Smart Contract (SC)s are a type of computer code stored on a blockchain, implementing a given logic. To some extent, one can draw an analogy between them and classes in programming language's literature. SCs, in technical terms, do not run on the blockchain. As blockchain is just a way to store data across all the participating nodes, it is used to distribute SCs across all the participating nodes. Because we are using a blockchain optimized for trans-

action management (refer Section 3.2.1.1), it is necessary to implement the concept of smart contracts separately for our proposed blockchain framework. To distinguish it from existent smart contract implementation, we refer our proposed implementation as Applied Smart Contract (ASC).

The process for ASC implementations starts by codifying a given logic. The given logic in our case is the DR allotment model. The new DR allotment model is first codified, compiled and then converted to byte code (to form a single binary) by the concerned TSO. This binary (ASC) is then subsequently encapsulated within a standard blockchain transaction. This is then added to the synchronized transaction pool by the TSO. CAs from tier 1 then certifies it by evaluating it on various criteria like fairness, time of execution, etc. Once certification is finished by CAs individually, the certified DR allotment model is then published on the blockchain (through consensus) and thereby subsequently stored across all the participating nodes.

During the DR allotment, when this ASC has to be executed, all tier 1 participants retrieve it and run it and verify the allotment in a distributed setting. At the end of the execution, a new block containing the output of the DR model (list of selected and rejected bids) is formed and added to the blockchain through consensus (similar to the process of adding a new DR allotment model on the blockchain). Each node verifies this block before adding it to its local blockchain copy. The consensus/verification is achieved knowing the fact that the same DR allotment model, replicated across all the tier 1 nodes, run with an identical interpreter, applied on the same data-set (common synchronized transaction pool), will yield the same output. We propose to use distributed consensus between identified block validators [Greenspan 2015] which works like PBFT but with a caveat i.e. single validator per block. Detailed execution steps of consensus using an example simulation are discussed in Section 3.2.4.

Even with the great similarity in the base concept, the proposed ASC differs from SC in many aspects. Starting with the language support, ASC can be in any language while SC has to be in the language supported by the blockchain platform. Because of this flexibility, ASC does not need an installation/maintenance of separate Virtual Machine (VM) unlike in the case of SC where running a separate VM is indispensable. For example, in the case of Ethereum, running Ethereum VM is absolutely necessary. Next, running a separate VM has a large overhead and sharing the live state of variables within the SC with participating nodes (using underlying P2P communication protocol) further increases it. As ASC doesn't need VM per se, the overhead for running the blockchain is significantly lower.

However, as VM is not necessary in the case of ASC, maintaining the live state of a variable in the code/ASC has to be managed separately using proper

indexing/retrieval mechanism if the need arises. To summarize, our proposed ASC offers most of the common features of the SC, i.e., immutable-persistent storage of data/code, trace of the transactions, trace of ownership of data, with much-reduced overhead because of the absence of a dedicated VM.

3.2.1.3 Escrow Accounts

In our framework, for DR settlement, we propose to use a decentralized escrow account setting. An Escrow account, in its simplest form, is an account where the asset is held in the interim till the transaction between the two parties is completed [Banton 2019]. The escrow account is directly controlled by an external third party which is not involved in the transaction under the hypothesis that it is unbiased, impartial and trustworthy. When escrow account setting is implemented on the blockchain, the trusted third party is effectively replaced by the blockchain itself thereby making the escrow setting truly decentralized and unbiased. Escrow account setting on a blockchain can be implemented in two ways i.e., 1) Using SCs (like LocalEthereum [LocalEthereum 2017] implemented on Ethereum Blockchain), 2) Through native support provided by specific blockchain platforms (like multi-signature wallets implemented in Multichain platform [Greenspan 2015]).

In our framework, we propose to use the escrow account setting implemented using “multi-signature wallet on the blockchain" as it is faster, lighter and simpler due to: 1) absence of an extra VM, 2) Native support on the blockchain platform. Also, in the case of SCs, the code needs absolute certification, as improper SCs implementation may lead to payment vulnerabilities (like the Decentralized Autonomous Organization (DAO) attack [Mehar *et al.* 2019]). In a decentralized escrow account based on the multi-signature wallet, the inflow of funds/tokens is unrestricted but the outflow is restricted by a constraint of multiple signatures. For instance, if a decentralized escrow account is owned by 3 entities on a blockchain, to make any outflow payment, 2 entities (majority) need to sign the same outflow transaction to validate the payment execution.

Further, an escrow setting based on multi-signature wallets does not have scaling, timing, and overhead issues because the payment decisions are done off-chain by the concerned entities and only the payment execution is carried out by each participant (involved in the transaction) on the blockchain. Further, it is truly decentralized as all participating entities in the transaction have an equal say. Figure 3.2 explains the typical DR settlement flow. In our framework, for implementing an escrow setting, we propose to use 3-signature wallets i.e., wallets jointly owned by three entities. For each participating ag-

gregator, there is a 3 signature wallet co-owned by that specific aggregator[1], the concerned TSO and the third party/CA. To withdraw the assets from these accounts, any 2 out of 3 signatures are required (TSO-Aggregator or TSO-CA or Aggregator-CA). The assumed hypothesis for the third party/CA is that it is trusted and unbiased. The problems arising in case of its misbehavior are out of the scope of this thesis.

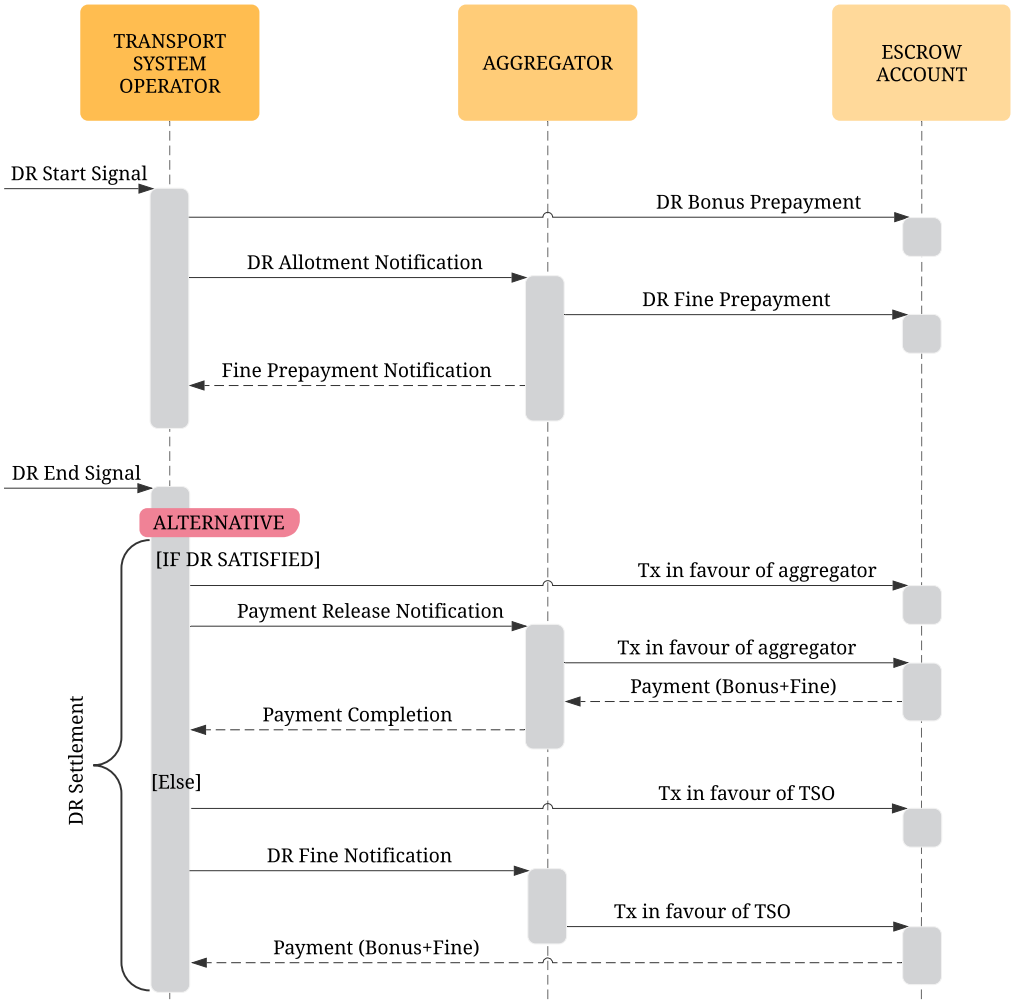


Figure 3.2: A typical DR settlement flow using the escrow account.

After the DR session starts, the DR preposition bids are collected by tier 1 from the tier 2 participants and the DR allotment model is run in a distributed setting. Once the subset of aggregators is selected, for each aggregator, the TSO deposits the bonus in advance in the respective escrow accounts corresponding to each aggregator. After the advance payment, the aggregator(s)

[1]separate wallets facilitates the easy financial auditing.

are asked to pay an advance fine in the same corresponding account in case if the aggregator(s) does not respect the agreed DR power consumption reduction. After the end of DR monitoring, a transaction containing the payment and the fine is then signed by TSO in favor of the aggregator if the allotted power reduction baseline is followed else in its favor. This transaction is then forwarded to the corresponding aggregator for signature after which the payment is done from the escrow account to the pertinent account. In case of discrepancy, either of the entities won't sign it and the decision then rests with the third party/CA which would do arbitration based on the data from the DR monitoring session.

3.2.1.4 Secure Element

In our framework, SE is used in signing the DR bids and in DR monitoring. It is used as a HSM add-on by the aggregators (see Figure 3.1). The SE stores the private key (used for signing the data) in its TSE. TSE ensures not only that the private key cannot be leaked/copied but also the authenticity of data in DR bids and DR monitoring. The corresponding public key is bound to the identity of the aggregator having that particular SE using Public Key Infrastructure (PKI) [Adams & Lloyd 1999] and then shared with the concerned parties (TSO, CAs). The decentralization of the DR session does not affect it as: 1)The process of identity binding is carried out for all new aggregators before they can start participating in any DR session, 2) The identity binding is carried out by appropriate registration authority and is point-to-point. Further discussion on the process of identity binding using PKI is out of the scope of this thesis.

When the DR session starts, aggregators decide on the amount of power consumption reduction to bid. This bid is then sent to SE for signature which is then ultimately transferred to Blockchain using BaaS through tier 1 participants. As the private key is stored in the SE, the tier 1 participants can have the surety that the bid indeed came from an authorized aggregator and is not changed during transmission[2].

For DR monitoring, we propose the use of a smart meter. This smart meter is connected to the SE of the aggregator (see Figure 3.1). When the smart meter records data (power consumption value), it is sent to the SE for signature. Once the data is signed, the authenticity of the data can then easily be trusted. This can be reasoned further by understanding the hardware integrity of the smart meter. The smart meter's hardware integrity is managed by the SE's tamper interrupt pin. Hence, when the smart meter is tampered, the SE detects it through its tamper interrupt pin and subsequently stops

[2]on successful verification of signature using corresponding public key.

Table 3.1: All possible cases of arbitration. Yes indicates the concerned party is able to provide signed data, No indicates otherwise.

Third Party/CA	TSO	Aggregator	Arbitration
Yes	Yes	Yes	Yes
Yes	Yes	No	Yes
Yes	No	Yes	Yes
Yes	No	No	Yes
No	Yes	Yes	Yes
No	Yes	No	Yes
No	No	Yes	Yes
No	No	No	No

signing the data. Considering this, when the signed data arrives from this smart meter, one could guarantee that the smart meter is not tampered. Further, on verifying the signature, the receiving entity can be sure that the data they received is of verified origin and unchanged during transmission (else signature verification would fail).

Furthermore, we propose that during the DR session, this signed power consumption data is stored by the aggregator (participating in the DR) in multiple copies (to be resilient against hard disk failures). The aggregator then transfers this data to the TSO (to enable it to make payment decisions in DR settlement) and to a trusted third party/CA. Further, in case of a dispute, the payments from the escrow accounts are made based on the audit of the signed data by the trusted third-party/CA. Given the architecture of the SE and hardware integrity of smart meter, a malicious party cannot generate fake data for the same timestamp range with proper signature thereby making the dispute-solving procedure reliable[3].This also ensures the arbitration in case of a dispute when at least one of the parties (CA/TSO/Aggregator) can provide the complete data.

Table 3.1 lists all the possible arbitration cases. It fails only if all the parties fail concurrently to provide the signed data. Next, we have an understanding that storing this signed power consumption data on the blockchain does not serve any additional benefit because this data is already tamper-proofed (by signing it using SE) and concerned parties will anyways store it for their protection of self-interest during a DR session. Moreover, as blockchain replicates everything on each participating node, this would generate unnecessary copies of this huge data-set(s) thereby causing scaling issues.

[3]similar concept used in chip and pin credit/debit cards to secure offline transaction.

3.2.1.5 Synchronization in the Framework

This subsection deals with clock synchronization and transaction pool synchronization in our framework. Starting with clock synchronization, during the DR session, tier 1 participants accept bids from aggregators (tier 2). These bids are only accepted between a given DR time window. If the tier 1 participants are not clock synchronized, then the gate closing time wouldn't be uniform across them. As aggregators are free to choose any tier 1 participant(s), clock synchronization is vital to avoid any unforeseen leverage (due to uneven gate closure time). As tier 1 participants are distributed and may in some cases belong to different geographical locations, an accurate time synchronization protocol is needed. Precision Time Protocol (PTP)v2 [IEEE Instrumentation and Measurement Society 2008] is one such protocol that can synchronize all the nodes within the sub-μs range. PTPv2 is a layer 2 protocol meaning nodes outside a given subnet cannot be clock synchronized. To resolve this, a GPS clock is added as a grandmaster clock on each tier 1 participant, which PTPv2 uses to synchronize timing on them. The effective clock accuracy with this combined approach is in sub-μs range as well[4]. This clock synchronization accuracy is acceptable considering that even real-time systems have sub-ms requirement.

Next, for the transaction pool, synchronization to have the same sorted order of all bids is imperative to ensure the same input and thereby the same output for the DR allotment model when running in distributed setting across all tier 1 nodes. The sorting mechanism for bids in the common transaction pool involves bid's hash. Here, we propose to use *Secure Hash Algorithm (SHA)-512* hash function. Bid with the lowest magnitude of hash is ranked first and the rest are sorted similarly. If an aggregator sends the same bid to more than one tier 1 participant, then in the common transaction pool, only one bid is kept (duplicates are removed). This ensures, for each DR session, there are only as many bids as the number of aggregators participating in that specific session. Further, In the case of hash collision i.e., the same hash for bids from two different aggregators, the bid with the lower hash value of *bid + mac address of the corresponding aggregator* is given precedence. As hashing is an unpredictable random function, fairness is thus ensured.

Further, to be deterministic with maximum delays while solving the DR allotment model (exponential time), there is a cap on the number of bids/aggregators on which the model is applied. Here again, the first n bids (thus, n aggregators considering 1 bid/aggregator) from the sorted synchronized transaction pool are considered. The proper limit on n is elaborated in Section 3.2.4. Independently, Tier 1 participants also time-stamp the bids

[4]GPS clock accuracy is in sub-μs range.

they receive. With this, there is a possibility that it is changed to favor a specific aggregator. To minimize the impact of this misbehavior, we propose to use time-stamp only for checking if the bid was received before the gate closure and not in the actual DR allotment model. Using this approach, it is impossible for any malicious tier 1 node to guarantee a favor by changing the timestamp for the colluding aggregator as the ordering is completely random (based on hash).

3.2.2 DR Allotment Modelling

Our framework is completely agnostic for the DR allotment model. Participants from tier 1 can plug in any DR allotment model in the framework which then has to be certified by CAs (see Section 3.2.1.2) before being utilized in a DR session. There are DR allotment models proposed in previous works like [Conejo *et al.* 2010] and [Chen *et al.* 2010], but they are adapted to micro-grid needs. For the completeness of our work and simplicity in understanding, we present a novel DR allotment model for TSO managed DR in the specific context of macro-grids. To determine an optimal DR allotment for the participating aggregators, the framework relies on a Mixed-Integer Linear Programming (MILP) formulation where integer variables are binary. More precisely, the MILP optimization tries to select a higher subset of aggregators while increasing the accepted fraction of their proposed bids (with respect to the DR signal), with corresponding higher bid amounts and reliabilities.

Each aggregator is assigned an average reliability value, computed by CAs/regulators and TSOs. The reliability of an aggregator defines its probability to satisfy a submitted bid. For example, a reliability of 50% for an aggregator means that it can, on average, satisfy half of its submitted bid. Initially, all aggregators have the same average reliability value, 100% which is then subsequently updated after each DR session by the TSO and is then stored on the blockchain (see Section 3.2.3). The total reliability of a subset of aggregators is the product of their reliabilities. Consequently, the logarithm of the total reliability is the sum of individual logarithmic reliabilities. The combined reliability for the selected aggregators should be greater than the TSO's required reliability. Thus, the MILP can be formulated as follows:

$$
\begin{aligned}
&\text{Maximize:} && \frac{\alpha}{D}\sum_{i=1}^{n} f_i b_i r_i + \frac{\beta}{n} \times \sum_{i=1}^{n} \lceil f_i \rceil \\
&\text{Subject to:} && \sum_{i=1}^{n} f_i \times b_i \leq D; \quad \sum_{i=1}^{n} \lceil f_i \rceil \times \log r_i \geq \log R; \\
& && f_i \in \{0\} \cup \left[\max\left\{ f_{i_{\min}}^{req}, f_{i_{\text{fair}}} \right\}, 1 \right], i = 1, \ldots, n; \\
& && \lceil f_i \rceil \in \{0, 1\}, i = 1, \ldots, n
\end{aligned}
\tag{3.1}
$$

Where:

- α, β are the scaling factors to balance DR value fulfillment and distribution;
- D is the total demand released by TSO;
- n is the total number of aggregators participating in a given DR session;
- R is the total reliability required by the TSO.

For aggregator i:

- b_i, r_i is the proposed bid of power consumption reduction and the avg. reliability respectively;
- f_i is a ***variable*** which denotes the fraction allotted by TSO to the aggregator i after optimization. Hence $f_i \times b_i$ is final power reduction to be done by aggregator i;
- $\lceil f_i \rceil$ is the ceiling of f_i implemented by introducing a ***binary variable*** i.e., $w_i = \lceil f_i \rceil = 1$ if $f_i > 0$, else 0;
- $f_{i_{\min}}^{req}$ is the minimum fraction requested for allotment by aggregator i;
- $f_{i_{\text{fair}}}$ is the fair fraction of allotment which ensures fair distribution of the DR signal and is given by:

$$f_{i_{\text{fair}}} = \frac{b_i \times r_i}{\sum_{j=1}^{n} b_j r_j}$$

As Equation 3.1 contains binary variables ($\lceil f_i \rceil \in \{0, 1\}$), it is categorized as Non-deterministic Polynomial time (NP)-complete problem [Karp 1972] due to the combinatorial explosion of its integer variables. Thus, for given n aggregators, there are at maximum 2^n possible distinct binary solutions where each solution is a subset of selected aggregators. There is no known

polynomial-time algorithm to find an optimal solution. The exact algorithms (branch-and-bound, cutting-planes) that find an optimal solution may need an exponential number of iterations. As TSOs require an optimal solution, we apply the well known branch-and-bound method [Land & Doig 1960], as it reduces the number of iterations by exploring a subset of feasible integer solution instead of all possibles solutions. Next, to be deterministic about maximum execution time, we propose a limit on the number of aggregators (n) for which the optimization algorithm will be applied (see Section 3.2.4).

3.2.3 DR Framework in action

The DR session starts when the TSO multi-casts the DR requirement $\boldsymbol{D}$ using Open Automated Demand Response (OpenADR) [OpenADR Alliance 2013]. All the subscribed participants in the tier 1/tier 2 get notified about it. The gate for bidding is then opened by all participants in tier 1 (see Figure 3.3). Interested aggregators in tier 2, send their secure signed (by SE) bid b_i to one or more tier 1 participants. This way, the acceptance of bids is completely decentralized and distributed thereby eliminating SPoF. When the gate closes, bids collected by all tier 1 participants are synchronized and sorted in the synchronized transaction pool so that every tier 1 participant has the same set of sorted bids.

Next, the certified DR allotment model (retrieved from the blockchain) is then applied on the common sorted bid list by each participant in tier 1. When the DR allotment is done and subsequent consensus among tier 1 participant is reached, the resulting DR allotment is written on the blockchain by adding a new block to the blockchain which contains all the bids but segregated into two groups viz., selected and rejected. This helps to maintain an immutable chronological history of the DR marketplace thus giving additional details such as the total energy bids accepted, current bidding trends, etc. This also makes it impossible for a malicious TSO to favor a particular aggregator as its proposed DR quota-distribution would be ruled-out during consensus.

Next, the communication between aggregator(s) and TSO is separate. Hence, delay by any aggregator to take required action does not affect the DR session for other remaining aggregators. Further, for each aggregator whose bid is accepted, TSO notifies the allotted percentage of its bid also pays in advance to the corresponding escrow account on the blockchain (see Section 3.2.1.3). This ensures that DR participants are paid if they follow the allotted power consumption reduction baseline. If the TSO diverges from this assumed hypothesis, a proper redressal mechanism is to be used whose discussion is out of the scope of this thesis.

Further, each selected aggregator is required to pay an advance fine in

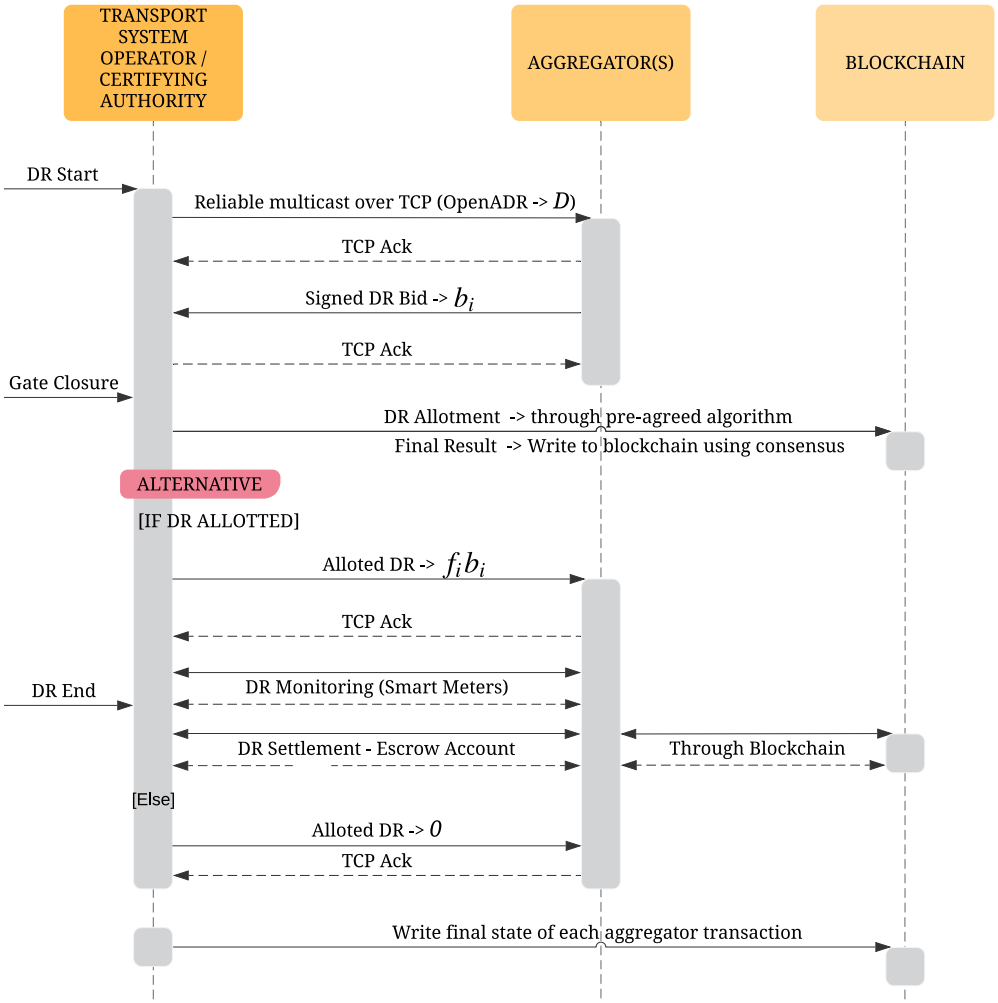


Figure 3.3: A typical DR session overview.

the corresponding escrow account (before the pre-agreed timeout) ensuring it is fined if it is not respecting allotted baseline. If the aggregator fails to prepay the fine before the time-out, it is not considered for the subsequent steps. DR monitoring starts only after prepayment from both sides. The DR monitoring is secured using smart meters and SE (see Section 3.2.1.4). After DR monitoring ends, for DR settlement, TSO makes the payment/deduction using the same escrow account through blockchain and if aggregator disagrees, an arbitration is opened with a pre-agreed trusted third party/CA. After DR settlement ends, the final state of each DR bid along with updated parameters like reliability, response time, etc., for the corresponding aggregator is written on the blockchain.

3.2.4 Simulation Results

The DR allotment model proposed in Section 3.2.2 was codified in python and used as ASC.Python PuLP library was used to solve the optimization problem. One example simulation is presented below. The required TSO parameters are given in Table 3.2 while parameters for individual aggregators are given in Table 3.3.

Table 3.2: Parameters for TSO.

Parameter	α, β	D (MW)	$R(\%)$	n
Value	1	100	0.0001	5

Table 3.3: Individual Simulation Parameters for 5 aggregators.

$Aggregator(i)$	**1**	**2**	**3**	**4**	**5**
b_i (MW)	100	100	100	100	100
$f_{i_{\min}}^{req}$	10%	10%	20%	25%	30%
r_i	10%	10%	20%	25%	30%

The final allotted DR quota (using the proposed DR allotment model) for each aggregator is shown in Figure 3.4. The blockchain consensus, in this case, is very important as that makes the whole DR allotment decentralized. Once the optimization is solved by the TSO, it shares the results in a standard blockchain transaction containing all the individual DR allotments. The dissemination happens using the blockchain P2P network maintained between all the tier 1 participants. One miner/verifier is chosen using the underlying consensus mechanism which will add this transaction to a new block after verifying the results in transaction by comparing it with locally obtained results. Then the block dissemination happens and every tier 1 entity similarly verifies the block before adding it to its local copy of the blockchain. Subsequently, tier 2 participants can access these results once they are on the blockchain making the whole process transparent.

Attack Model: Consider, TSO colludes with aggregator 1 to favor it with more allocation, as presented in Figure 3.5. The presented DR quota satisfies all the constraints of Table 3.2 and 3.3 but still favors aggregator 1 with more DR quota. The TSO subsequently creates a standard blockchain transaction containing this allotment scheme. However, during the consensus process, the transaction will be invalidated because it won't match with locally obtained results of other tier 1 entities making the allocation null and void. Instead, the proper allocation (Figure 3.4) will be validated as long as the majority of the network ($> 50\%$) validates it.

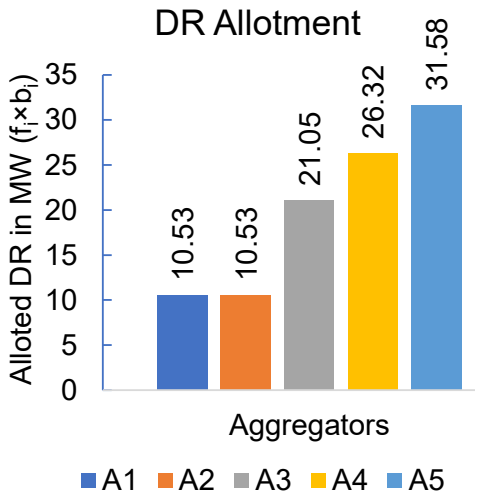


Figure 3.4: DR allotment using the proposed model.

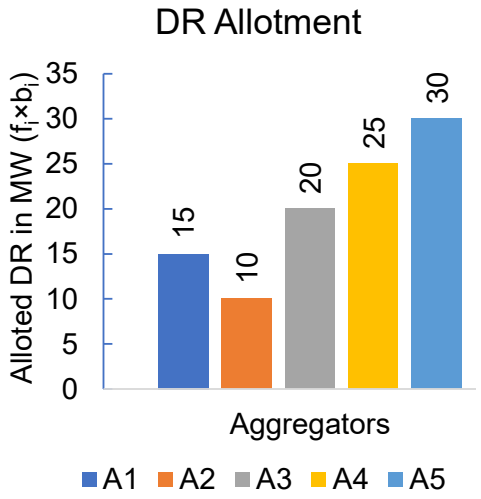


Figure 3.5: Biased DR allotment by TSO favoring aggregator 1.

Further, the performance of the DR allotment model proposed in Section 3.2.2 was evaluated against a wide range of simulated inputs and the results shows the effect of increasing the number of aggregators against various chosen system reliabilities in four areas, i.e., 1) Percentage of accepted bids, 2) Percentage of accepted aggregators 3) Average percentage allotment for selected subset of aggregators and 4) Time required to complete the allotment. For simulations, parameters from Equation 3.1 are set as given in Table 3.4 for TSO and in Table 3.5 for aggregators. The averaged results for 1000 iterations are presented next.

Table 3.4: Parameters for TSO.

Parameter	α, β	D (MW)	$R(\%)$	n
Value	1	100	$5, 25, 50, 75, 95$	$[10, 50]$, step=10

Table 3.5: Parameters for aggregators.

Parameter	b_i (MW)	$f_{i_{\min}}^{req}(\%)$	$r_i(\%)$
Value	$random(0, 50)$	$random(0, 100)$	$random(0, 100)$

Figure 3.6 shows the percentage of fulfilled DR. It is evident that irrespective of the value of R, the fulfillment percentage increases due to an increase in n which subsequently increases participants with required R. However, higher R requirement (= 95%) can reduce it to meagre 29% (on average, for lower n). Figure 3.7 shows that the % of selected aggregators is greatly reduced

when total participating aggregators (n) is increased as the demand D stays constant. The negative slope (< -1) of curves indicates that indeed the proposed MILP finds a delicate balance between distributing the demand and respecting the minimum allotment.

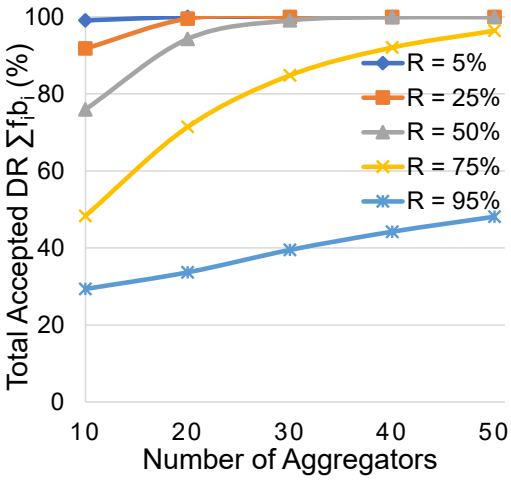


Figure 3.6: Total accepted DR (percentage of fulfillment).

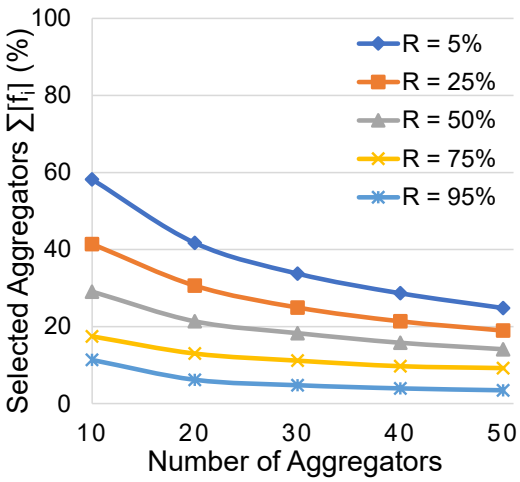


Figure 3.7: The average percentage of selected aggregators.

Figure 3.8 shows that the avg. allotted fraction f_i gradually reduces with increase in n or decrease in R. This is because both trends (increasing n, decreasing R) increase the number of actual aggregators being considered and hence the demand D gets divided. Next, Figure 3.9 depicts the execution time taken by the DR allotment model for various values of (n, R) which shows that it is directly dependent on the actual number of aggregators under consideration and can be reduced by as much as 82% by increasing R from 5% to 95% (on average for higher n). Further, for the lower value of R, the execution time rises more drastically ($\approx$ 4.5x) for an increase in n.

To summarize, with the proposed DR allotment model, a very low or very high value of R drastically affects some or the other metrics of DR allotment. Owing to this and to be deterministic about maximum execution time, we propose a suitable limit of $n = 20000$ as the MILP for it is solved within 55 seconds on i7-8550U CPU (1.80GHz). Considering real-world application, $n = 20000$ aggregators limit is very high, thus, making it apt for future scaling needs. Subsequently, the inter-block time for block formation in the underlying blockchain should also be higher than 55 seconds (e.g. 2 minutes).

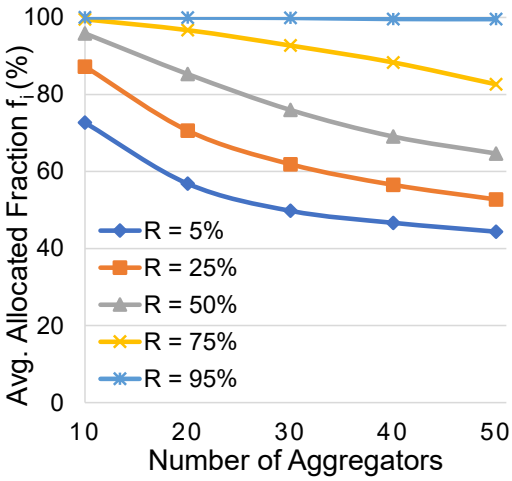


Figure 3.8: Average allotted fraction (f_i) for selected aggregators.

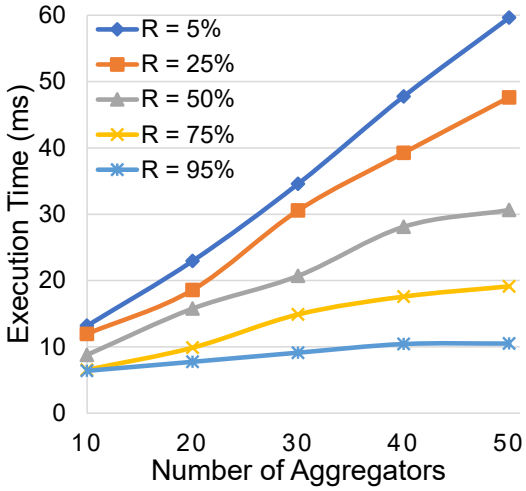


Figure 3.9: Execution time taken by the DR allotment model.

3.3 Combinational Approach applied to Smart Vehicles

Synopsis: In the context of Intelligent Transportation System (ITS), smart vehicles have become increasingly complex. Electronic Control Unit (ECU) within smart vehicles are now capable of performing intricate operations which ensures reliable functioning of the smart vehicle even in emergency situations. However, these ECU(s) lack a TEE and TSE. This makes it vulnerable to many security issues. To solve this, we apply our combinational approach based **Sa**feguarding **F**rame**w**ork (SaFe) [Deshpande *et al.* 2019c] for smart vehicles in which ECU(s) use SE for TEE and TSE. We justify the use of blockchain by showing how it securely facilitates application management on SE when ECU needs are changed. Leveraging on SaFe, we introduce the concept of *non-repudiable responsibility*. We present our realized framework and testbed based on NXP IMX6Q, Multos M5-P19, MultiChain. We show through our experimental results that how SaFe improves the performance of safety-critical operations within ECU by as much as 85%, all this while guaranteeing increased security, tamper-proofness, immutability and reduced memory, storage, processing overhead.

3.3.1 SaFe Conception

Because of the increased complexity of the smart vehicles, the ECU within is now designed in such a way that it can carry out very intricate actions triggering numerous other activities. The intra-vehicular system has completely

changed from being homogeneous (one ECU and few sensors) to complete heterogeneous (multiple ECUs with predefined hierarchy and numerous sensors). Following this, many interesting works have been proposed to improve the smart vehicles. However, there are few issues which still needs to be solved in order to make these smart vehicles more holistic and more secure with the activities they undertake.

Like, for all critical operations happening within an ECU, cryptography plays an important role to secure them. For example, encryption ensures privacy, signature ensures immutability as well as ownership of data, signed hash provides easy way for verification of data. However, these applied concepts work with one single basic assumption i.e. secure keys. Especially, in encryption and signature algorithms, the security of underlying keys is of utmost importance. The security of keys is not only related to their storage but also to their generation [Van Bulck *et al.* 2018]. Further, the environment within ECU also needs to be secured for the reliable implementation of encryption and signature algorithms [Kocher *et al.* 2018],[Lipp *et al.* 2018].

These issues related to security of key generation, key storage and execution environment can be solved with the proper implementation of TEE and TSE within the smart vehicles. We achieve this through our proposed SaFe. SaFe is a holistic framework which not only uses SE as a HSM for providing TEE and TSE for the smart vehicles but also provides a blockchain based subroutine for updating the code/keys within SE in a secure, trusted and decentralized way. It goes further by introducing a concept of *non-repudiable responsibility* by leveraging the blockchain and SE technologies. (Refer subsection below)

3.3.1.1 SE for TEE and TSE

Because of the secure by design approach, SE is an ideal technology for implementing security critical operations. With the aid of the internal crypto co-processor, SE can generate secure keys. Once they are generated, they are stored within the SE (secure storage). These same keys are used for encryption and signature which are also carried out within SE (secure execution). As the keys never go out of SE, they are secured in true sense. On top of this, there is a tamper-pin present in SE which detects any hardware tampering attempt and blocks any further operation.

3.3.1.2 Blockchain based Update-Subroutine

Generally, SE can store multiple codelets. Each codelet can store specific keys[5] and performs a specific task within the SE. These codelet, like other applications, are improved and updated over time by their respective developers (e.g. key updates, parameters update for signatures). Further, on introduction of a new feature, SE compatibility is needed requiring new public key installation on SE. This is done by issuing new codelet containing the new public key for the SE. However, to update/install codelet on SE is an arduous task. Multos M5-P19, for example, depends on special programming certificate called ALC for uploading the specific codelet to the SE.

This ALC cannot be generated by the end-user. In this case, the vehicle has to first request Application provider to get the signed codelet (Application Load Unit (ALU)) and then subsequently request the delegated CA for the ALC corresponding to the specific ALU [MAOSCO Limited 2017]. As both these procedures are point-to-point and centralized, it is an arduous task to guarantee high availability of centralized server across different time-zones and to properly provide required ALUs and ALCs for millions of vehicles according to vehicle's SE version, architecture etc.

Section 3.3.3 explains how SaFe solves this problem by proposing a Blockchain based update-subroutine. SaFe leverages on Blockchain's immutability and distributed availability thereby eliminating SPoF and also serving additional benefit of maintaining an immutable record of the delegation of programming rights (i.e. ALC) for SE within ECU(s).

3.3.1.3 Non-repudiable Responsibility

In context of autonomous vehicles, it is the vehicle itself which manages everything (including driving). With the use of SE, critical logs (i.e. after high impact accident, ECU malfunction) of the vehicle state can be securely signed to avoid tampering while in transmission and ultimately pushed to blockchain for immutable storage. This would make the regulators/investigation agencies carry out an audit in a meaningful way given that they can fully trust that the critical log was indeed generated by the vehicle in question.

This is important because it introduces the concept of non-repudiable responsibility for manufacturers (for ECU failures)[6], for developers (for malfunctioning machine learning algorithms), for insurance companies (for denying lawful insurance claims), etc.

[5] keys are stored in codelets' binary which is then uploaded to SE

[6] The clue for this is taken from recent Boeing 737-Max stalling problem [Aircraft Accident Investigation Bureau 2019]

3.3.2 SaFe Architecture

Our SaFe (see Figure 3.10) consists of 1) ECU containing the SE, 2) Blockchain and 3) ALU (signed codelet) issuing authority 4) ALC (programming certificates) issuing authority. ECU is the equipment within the smart vehicle which is capable to communicate with the Blockchain and acts as an intermediary between SE and Blockchain. For our framework conception, we use the popular IMX6Q-SABRESD board [NXP MX6Q] by NXP Semiconductors which is quite widely used for ECU implementation. It is worth mentioning that this selection is only for the proof of concept. SaFe design idea is platform agnostic and can be well adapted for other ECU platforms. Further, we use Multos M5-P19 [Multos form] as SE and MultiChain[Greenspan 2015] as blockchain.

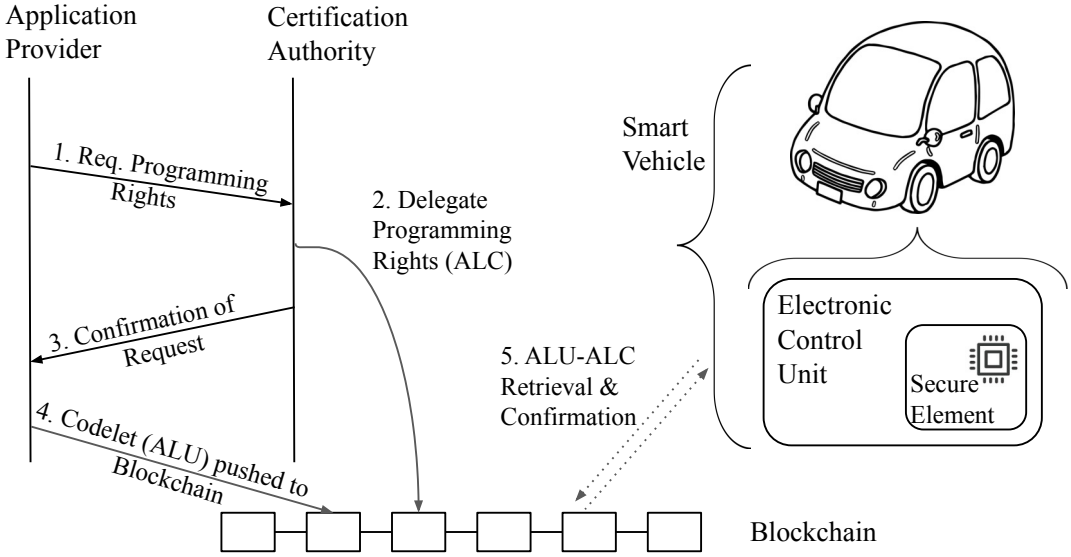


Figure 3.10: SaFe illustrating Blockchain based update-subroutine for Secure Element

IMX6Q (ECU) within the smart vehicle is connected to the Multos M5-P19 via serial connection. The M5-P19 (SE) contains the public keys of the various information providers like weather, traffic, etc. Whenever critical data is to be verified on ECU, these public keys are used to verify the signed data. Verification of the signature takes inside M5-P19. It also contains the private keys whose corresponding public keys are with the concerned parties which use them to verify data sent from the smart vehicle in a tamper-proof way like maintenance logs, software crash reports. The signing of these logs/reports also takes place within M5-P19. For all signature and verification purposes in SaFe, we used ECDSA[Johnson *et al.* 2001][7] while Elliptic Curve Diffie Hell-

[7] we used ECDSA because it provides higher level of security for smaller length of keys

man (ECDH)[McGrew *et al.* 2011] is used to derive shared secret key in a secure way for encryption.

The IMX6Q is connected to the different service providers through internet using the on-board SIM/Ethernet slot. IMX6Q is also connected to blockchain via internet. Here, it uses blockchain as a service instead of running a blockchain node on board as it is neither feasible nor reliable given that the smart vehicle frequently changes the position and internet bandwidth is constrained. The blockchain is maintained by different nodes in the cloud network which are owned by different entities like manufacturers, regulators, service providers. For our conception, we experimented only with one blockchain for all purposes. However, there can be more than one depending on the use-cases. For example, there can be a separate blockchain (maintained by manufacturer and service providers) dealing with codelets updates and another blockchain (maintained by regulators and certifiers) where critical log is stored. For securing hardware integrity, the tamper pin of M5-P19 is connected to the tamper detection circuit. On breach, M5-P19 ceases all operations and transmits a *revoke certificate* to IMX6Q which then publishes it to blockchain. The smart vehicle owner can then be informed and the manufacturer can then subsequently schedule a maintenance/repair for the smart vehicle in question.

3.3.3 Update Subroutine

With reference to Figure 3.10, the updated/new codelet is first developed by the Application Provider then compiled with all appropriate libraries linked to create a single executable binary which can be directly installed on SE. This binary is then extensively tested. If passed, it is then signed by the Application Provider to create an ALU binary. For privacy and security, this ALU binary can also be encrypted before being pushed onto the blockchain by using the public key pertaining to the specific secure element. For this, each SE publishes a public key on the blockchain which would then be used by the Application Provider for encryption.

This encrypted ALU binary is subsequently pushed to the MultiChain blockchain after approval from the CA. The approval is finalized through the delegation of rights for programming the SE on the blockchain. Here, the CA issues an ALC corresponding to the ALU and w.r.t SEs for which they are designated. Ultimately, the ALC contents determine for which SEs, the ALU is designated.

The algorithm for this is given in Algorithm 2. The process begins at the

*compared to RSA [Rivest *et al.* 1978], Digital Signature Algorithm (DSA) [Kravitz 1993].

CA's end on receiving the request from the application issuer. The request contains the header of the ALU and the public key of the application issuer and meta data. The CA then signs this to create a corresponding ALC. The ALC is then converted to hex string (for practical purposes). This hex string is then encapsulated within the OP_RETURN field of a standard empty transaction of MultiChain. This transaction is then sent to be included in the blockchain.

The next thing remaining is the publication of ALU on the blockchain. To realize this, in SaFe, we use Algorithm 3. To push the ALU on the blockchain, first the ALU standalone binary is converted to a binary string which is then converted to hex string and then subsequently published on blockchain in the similar way as corresponding ALC along with information of the target SEs and blockchain transaction id of ALC.

This process relates closely to publication of smart contracts (which are essentially program codes) on blockchain albeit with a caveat that the code (ALU) published on the blockchain can only be used for SE and not for the participating nodes. Once the new/updated ALU is on the blockchain, it is available for all compatible SEs of smart vehicles guaranteeing secure distributed access and hence mitigating the SPoF.

Apart from eliminating SPoF, the delegation of programming rights on the blockchain serves additional dual purpose. Firstly, it allows to securely program a remote SE. Secondly, it allows to have, an immutable record of the history of past delegations in case of disputes.

At the vehicle's end, the ECU periodically queries the blockchain and upon finding a suitable ALC-ALU, it downloads it and then installs/replaces the ALU on the SE using the ALC via serial communication. This installation is done at a suitable time (i.e. when the vehicle is not driven). The ECU can then confirm the update by issuing a transaction containing relevant information (id, update time, ALC/ALU transaction id) signed by the SE to the Blockchain [Urien 2018]. Once on the blockchain, these transactions serve as an important track of the SEs who have already updated the codelets and who hasn't. This record has to be immutable so that security vulnerabilities does not occur by roll-back of updates. Here, the blockchain works perfectly as a distributed shared and immutable database.

The entire update subroutine can work even by replacing the blockchain with decentralized server mechanism but it only solves the problem of distributed availability. However, the guarantee of immutability for maintaining 1) the update state of the SEs, 2) history of the delegation of programming rights and 3) Non-repudiable Responsibility (see section 3.3.1.3) is unique to blockchain and hence indispensable in this case.

Algorithm 2 Algorithm for delegation of programming rights (ALC) on blockchain

```
Data: Request from application provider
Result: Publication of ALC on Blockchain
begin
    initialization;
     Receive: ALC issuance request containing signed codelet header and
     public key;
     while ALC not published on blockchain do
        examine current request;
         verify public key of the application provider;
         verify signature of the codelet header;
         if success then
            go to next section;
         else
            quit replying "UNAUTHORISED" error message;
         end
         extract SE list from the request;
         extract corresponding keys from DB;
         generate ALC for each SE in the list;
         verify all the generated ALC(s);
         if success then
            go to next section;
         else
            quit returning list of SEs not found in DB;
         end
         hexlify (binary to hex string) all ALC(s);
         create blockchain transaction(s);
         insert hexlified ALC(s);
         connect to the blockchain;
         publish the transaction(s) to the blockchain;
         if success then
            quit showing success;
         else
            restart or quit based on user input;
         end
    end
end
```

Algorithm 3 Algorithm to push codelet to blockchain

```
Data: Standalone codelet
Result: Codelet Publication on Blockchain
begin
    initialization;
     Get: standalone codelet binary;
     while codelet not published on blockchain do
        test current codelet binary;
         if success then
            go to next section;
        else
            quit displaying linking error;
        end
        sign current codelet;
         encrypt current codelet;
         verify signature and encryption;
         create ALU with current codelet;
         hexlify (binary to hex string) current ALU;
         if success then
            go to next section;
        else
            restart or quit based on user input;
        end
        create a blockchain transaction;
         insert current hexlified ALU;
         connect to the blockchain;
         publish the transaction to the blockchain;
         if success then
            quit showing success;
        else
            restart or quit based on user input;
        end
    end
end
```

3.3.4 Performance Analysis

In SaFe framework, two main components i.e. Blockchain and SE are used in conjunction with the ECU of smart vehicle. We analyze impact of both components separately for simplicity reasons. Starting with blockchain, the impact on ECU w.r.t. memory, processing, timing is negligible in magnitude because blockchain is used as a service and nothing changes for the ECU as it only writes/reads data on blockchain with which it is concerned.

However, by empirical analysis, the use of blockchain does reduce the processing overhead on ALU and ALC issuers by a factor f:

$$f = \frac{\text{No. of Smart Vehicles}}{\text{No. of types of Smart Vehicles}}$$

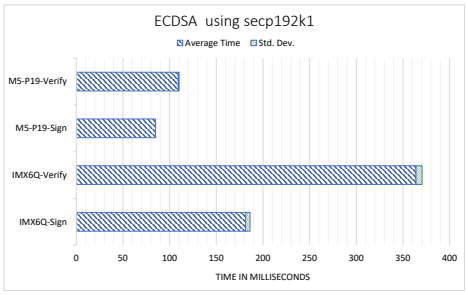
with many other additional benefits like immutability, non-repudiation and decentralization but at the cost of increased storage overhead by a factor S:

$$S = \text{No. of Nodes maintaining the blockchain}$$

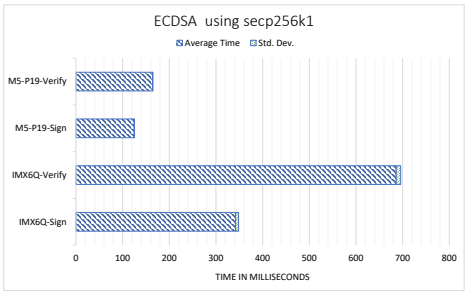
To analyze the performance and overhead of the SE in the SaFe, we compare two distinct scenarios i.e. Scenario 1: ECU without SE and Scenario 2: ECU with SE. In scenario 2, as the ECU delegates all the critical operations of signing, verification, generation of keys/storage of keys to SE, the overhead of memory, computational processing on the ECU actually decreases (compared to scenario 1) as ECU no more generates the computationally intensive signatures or keys and nor it does stores the keys.

On the timing axis, it gains the time as SE is much faster in executing cryptography operations. To put it numerically, we implemented a testbed. IMX6Q was used as ECU. Ubuntu 18.04.2 (minimal) along with Kernel 4.9 Long Term Support (LTS) (Real Time) clean install was performed on IMX6Q. A script implementing ECDSA signing and verification was developed on it. For the SE, M5-P19 was used. A similar script was created for the M5-P19.

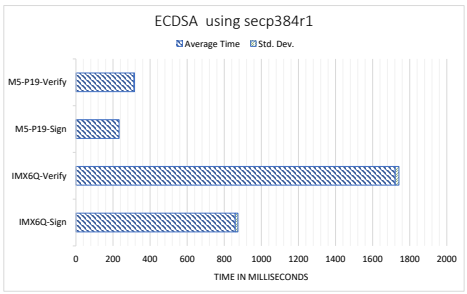
Raw keys and domain parameters corresponding to secp192k1, secp256k1, secp384r1, secp521r1 curves were generated in hex format using openssl separately beforehand for the scripts. For evaluation, a random QWORD was generated on both scripts. This was subsequently hashed using SHA512 algorithm (As it is a standard and recommended practice to sign only the hash of data). Federal Information Processing Standards (FIPS)180 [FIPS 2012] truncation was applied (when hash bit length was greater than curve bit length). Then time to ECDSA-sign and ECDSA-verify were recorded using respective timer functions of both scripts. This process was repeated 100 times on both ECU and SE for all the 4 curves.



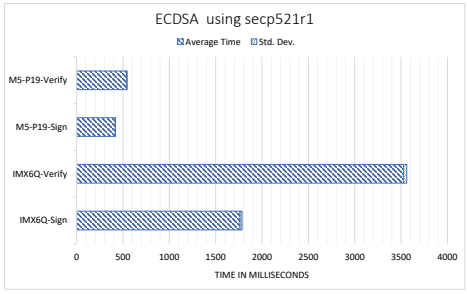
(a) n = 192 bits; $hash$ = 512 bits, FIPS180 = yes



(b) n = 256 bits; $hash$ = 512 bits, FIPS180 = yes



(c) n = 384 bits; $hash$ = 512 bits, FIPS180 = yes



(d) n = 521 bits; $hash$ = 512 bits, FIPS180 = no

Figure 3.11: ECDSA Performance Evaluation, IMX6Q vs M5-P19

Average of the timing results along with standard deviation is shown in the Figure 3.11. In general the time taken by both devices was directly proportional to the curve length. i.e. longer the curve, more time they took. Further, verification took longer than signing and results were more drastic (2x) in case of IMX6Q. Performance gap between the two broadened with the increase in the curve bit length. Also, IMX6Q was less deterministic compared to M5-P19. On an average, SE took 53%-76% less time for signing and 70%-85% less time for verification compared to IMX6Q making the whole ECDSA implementation atleast 2x-5x times faster. The only overhead, which SE impacted negatively was cost. For an EAL7 certified SE, extra 10€ cost is added (economy of scale would reduce it further). But with this negligible cost overhead, SE brings unmatched safeguarding to the whole ECU and smart vehicle ecosystem.

3.4 Combinational Approach applied to IoT

Synopsis: In the context of the IoT, the issue of holistic data security is complex. A reliable automatic decision with high accuracy can be taken only if the sensor data is secured at 3 critical points i.e., 1. Point of generation (IoT sensors), 2. Point of storage, 3. Point of usage (IoT actuators). This is a tricky task as IoT devices/sensors are constrained with limited processing power, memory, battery autonomy, storage. Any additional implementation of cryptographic functions for increasing security severely affects their performance. Further, in the IoT domain, the environment is completely heterogeneous making it extremely difficult to implement single security protocol across all remote entities.

Nonetheless, the problem has been solved *partially*. Point 2 is adequately addressed by Blockchain that offers remarkable immutability by storing data securely and transparently in a distributed setting, making it tamper-proof while in storage. However, blockchain alone cannot root out all the data security impediments in an IoT network as Points 1 and 3 still needs to be resolved. To solve this, we apply our combinational approach and propose a novel **S**ecure **E**lement **B**lockchain **S**tratagem (SEBS) [Deshpande *et al.* 2019a] to effectively secure all 3 Points at once, thereby realizing light, holistic, efficient IoT data security in true sense. Further, in a generic IoT network scenario, we show how it improves the performance of critical security operations by as much as 31 times.

For simplicity, we segregate use-cases in IoT into two types of scenarios and explain their distinct impediments:

3.4.1 Scenario 1: Blockchain as Data Sink

In this scenario, blockchain is the last component in the data flowchart i.e. it acts as a data sink. Typically, for this scenario, data is collected through Wireless Sensor Networks deployed on the remote IoT site and subsequently stored on the blockchain for future use (decision/policy making, automatic payments). A typical application of this scenario can be found in smart home use-cases that secure sensor data on blockchain. In this, even if the data is secured after storing on blockchain, the authenticity, tamper-proofness and origin of the data cannot be verified effectively as data is not secured at the generation point. The principle data aggregator from the remote IoT site aggregates all data from sub-aggregators and push it on the blockchain. As there are no measures to detect tampering, authenticity at sub-aggregator level, this data is effectively used as it is without any additional safeguards. These additional safeguards are important in some use-cases like in renew-

able energy smart-grids where payments directly depend on the output of the sensor values.

3.4.2 Scenario 2: Blockchain as Data Source

In this scenario, blockchain is the first component in data flowchart i.e. it is used as a data source. Typically, for this scenario, blockchain is used as a distributed secure immutable storage serving as a backbone database. The data is retrieved from this backbone database and certain actions are performed on the remote IoT site depending on the data value/state. blockchain-based access control systems are an apt example of this scenario. For this, even if the data is secured through its storage on blockchain, the end entity i.e. Remote IoT Device (RID) in this case depends on an intermediary for its retrieval and also lacks the resources needed to verify this data. This can have critical security ramifications in use-cases like blockchain-based firmware update systems where the end-point retrieves data from blockchain as a service and lacks resources to ascertain if authentic data is coming from the blockchain.

To summarize, the challenges from both scenarios need to be solved effectively to completely secure IoT data. Further, it becomes even more intricate, given the limitation of RIDs which are severely resource-constrained. In section 3.4.3, we explain how SEBS fill this exact niche.

3.4.3 The SEBS

The main purpose of SEBS is to secure the data transactions at 3 points i.e. 1) point of generation, 2) point of storage and 3) point of usage. For this, SEBS leverages on SE and blockchain. Scenarios 1 and 2 (see Section 3.4) requires to secure point 1-2 and 2-3 respectively for meaningful data security in IoT. To avoid redundancy in explanation, we illustrate SEBS implementation w.r.t Scenario 1 and highlight the differences w.r.t. Scenario 2.

3.4.3.1 SEBS implementation in Scenario 1

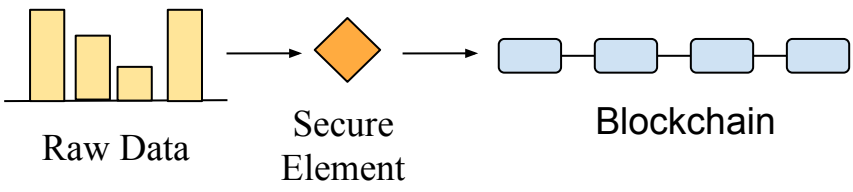


Figure 3.12: The conceptual SEBS framework (data-flow).

The SEBS conceptual framework for Scenario 1 is presented in Figure 3.12 which describes the data-flow. The concept behind it is that the data (raw) is passed through SE for tamper-proofing before it is stored on the blockchain. The implementational framework for the same is presented in Figure 3.13. The process to use SEBS starts at the sensor level. The sensor can be either directly connected to the SE or through sub-aggregator, depending on the type of the sensor and required granularity in security. The raw data from the sensor(s) is sent to the SE. The SE then subsequently signs the data using the private key stored within it. This private key never leaves the SE and nor it can be copied (owing to “secure by design" architecture of SE).

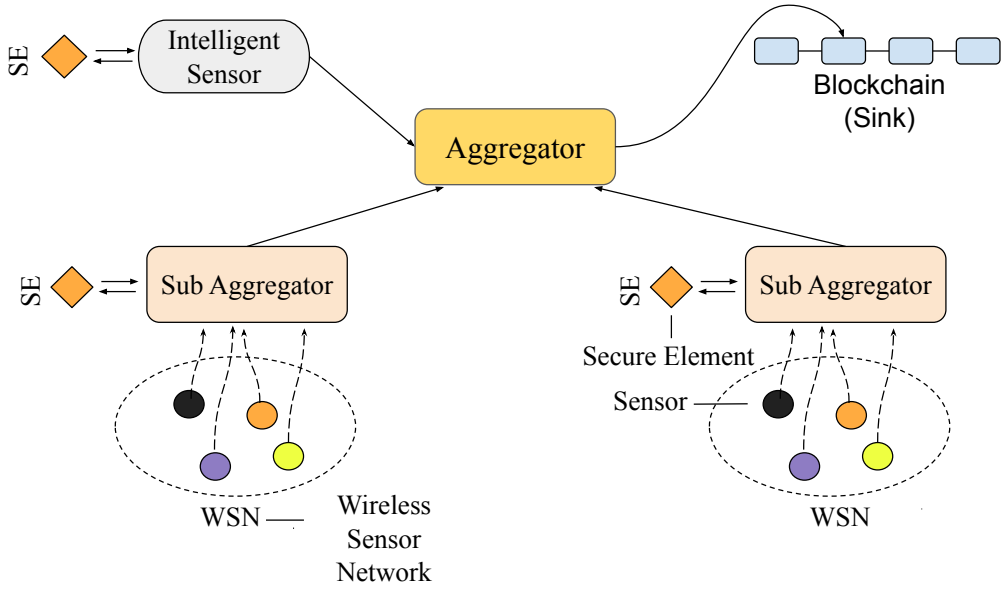


Figure 3.13: SEBS implementation when blockchain used as a data sink (Scenario 1).

Once the endpoint receives this signed data and verifies the signature, it can be sure that indeed the data came from a particular sensor/sub-aggregator with whom this SE was associated. Further, if privacy is needed, the data can be encrypted with SE. Next, with SE, hardware integrity can be guaranteed as well, using a tamper detection circuit. If any tampering attempt is done, this circuit makes the tamper detection pin on SE high, following which the SE ceases all its functioning. Next, the data is then sent to blockchain (via sub-aggregator and/or aggregator) for secure and persistent storage. As the data is signed, it ensures the authenticity and integrity of the data while in transmission.

The end-point, which verifies this data during an audit, can, therefore, be assured about its stated place of origin, authenticity, and integrity during

generation and transmission (point 1) and reading it from the blockchain verifies that no part of the data was deleted after storage (point 2). Thus, data obtained under SEBS cannot be denied/disputed by any of the concerned parties as whole process from data generation point to data storage point is secured. In audits, this helps to enforce the concept of *non-repudiable responsibility*. This applied concept is very useful in the context of IoT. The culpable party cannot deny the validity of data authenticity, integrity, origin, etc., on technical grounds during litigation, thereby resulting in faster dispute settlements.

3.4.3.2 SEBS implementation in Scenario 2

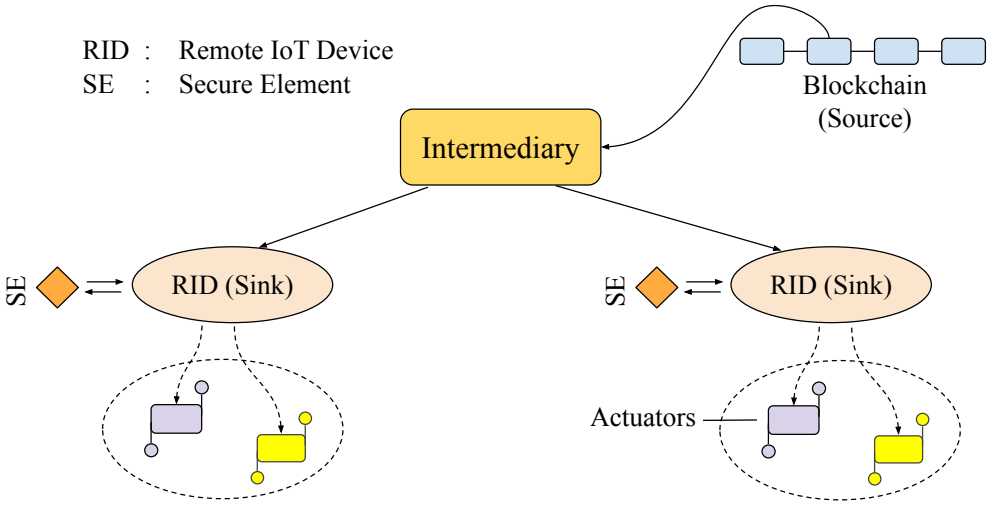


Figure 3.14: SEBS implementation when blockchain used as a data source (Scenario 2).

The implementation of SEBS in Scenario 2 is very similar to Scenario 1 except that blockchain here is used as a data source while the RID is used as a data sink. The RID, in this case, the sub-aggregator (see Figure 3.14), receives data from the blockchain through an aggregator. Given blockchain's immutability, the data is retrieved in its entirety (point 2). On reception, the data (signed) is sent to SE for signature verification and/or decryption. On success, the data origin and authenticity are thus ensured (point 3).

3.4.4 Overhead Analysis

In SEBS, the novelty lies in the additional intelligent usage of SE with traditional blockchain-based data security approaches. Since SE is installed as an

additional component at the RID, we perform overhead analysis at RID.

3.4.4.1 Computational Overhead

RIDs, being severely constrained, are either incapable or acutely disadvantaged to perform resource-intensive cryptographic operations securely. With SE as an HSM add-on, they gain this ability at a cost of increased computational overhead for maintaining communication with SE (serial/I2C/SPI). This little (increased) overhead cost is still significantly lower against the native implementation of cryptographic functions.

3.4.4.2 Timing Overhead

To measure the timing overhead, we set up a testbed consisting of RID, blockchain node, and a SE. We used Arduino Nano 3.0 and Raspberry Pi 2 Model B as RIDs. For SE, we used Multos M5-P19 [Multos form]. To quantize the timing overhead, we performed 100 iterations of ECDSA signature and verification on each RID and SE. The averaged results with standard deviation are presented next.

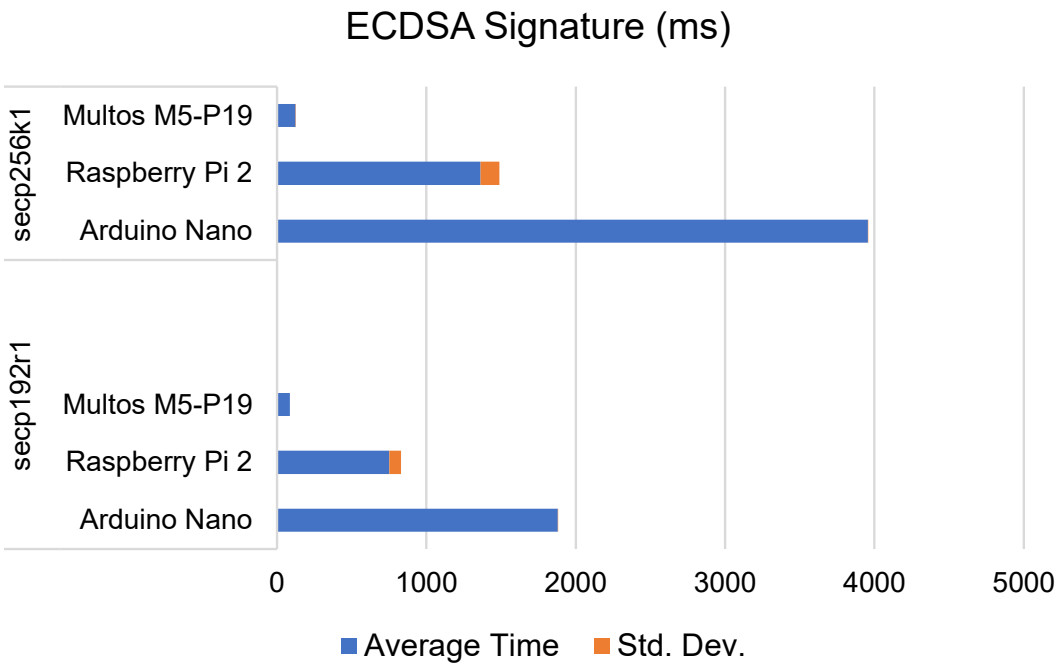


Figure 3.15: ECDSA signature time cycle, RIDs vs SE.

Given its secure architecture and specialized crypto-processor, SE performed much better compared to RIDs. The performance improved further for bigger ECC curves. For ECDSA signature (Figure 3.15), Multos SE was

up to 31x and 10x faster compared to Arduino Nano and Raspberry Pi respectively. Similarly, for verification (Figure 3.16), it took up to 25x and 16x less time respectively (secp256k1).

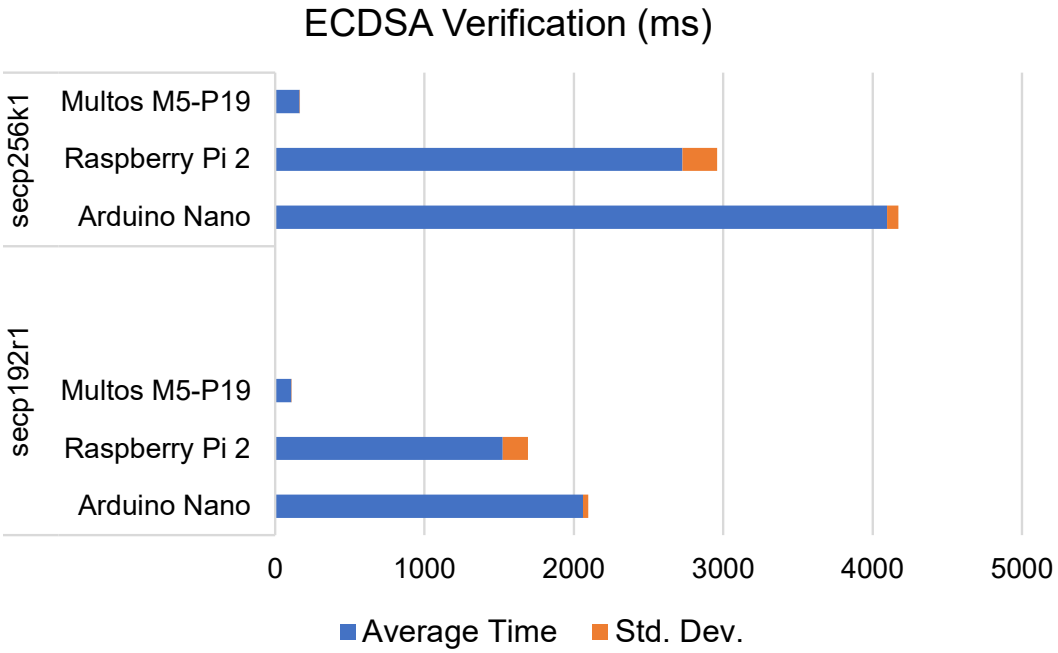


Figure 3.16: ECDSA verification time cycle, RIDs vs SE.

When combined, the equivalent timing overhead of the SEBS is reduced as the timing overhead at the RID level is significantly reduced in magnitude compared to the native implementation of ECDSA on RID.

3.4.4.3 Memory Overhead

Assuming we use ECC [Johnson *et al.* 2001] with a 256-bit curve, at RID level, one private key of 32 bytes for signature and one public key of 64 bytes for verifying data's blockchain affiliation is needed. As these keys are stored in SE (TSE), the memory overhead is reduced by 96 bytes. Although with a small factor, the SE reduces overall memory overhead.

3.4.4.4 Cost Overhead

Since additional hardware component installation is required at the RID level, the cost overhead is increased. The cost of SE varies with the certification level. The cheapest available SE (non-certified) costs 0.5€ while Evaluation Assurance Level (EAL)7 certified SE costs 10€. Since, in our proposition of SEBS and SEOVA, one SE is needed at each RID end (source and sink), the

cost overhead is between 1€ to 20€ depending on the required certification level. In a nutshell, our proposed SEBS reduce computational, timing and memory overheads greatly in a given IoT system. Further, when gauged against the benefits, they clearly outweighs the little increased cost overhead.

Summary: In this chapter we presented the combinational approach using blockchain and SE. We applied this approach to 3 disparate fields and evaluated the performance. However, the niche issues of data affiliation and sensor monitoring (system liveness) are not solved by this combinational approach. As a one step further, in the next chapter, we explain how these niche issues can be solved with the use of SE, thereby realizing holistic transaction security in true sense.

Chapter 4

Certifying Data Affiliation and System Liveness using SE

Contents

4.1 Introduction

In Chapter 3, the proposed combinational approach solves nearly all the problems of holistic data security in IoT. However, there are two classical problems related to IoT which need to be addressed separately. The first problem relates the problem of data affiliation when blockchain is used for secure storage of data. The second problem relates to the sensor monitoring part in IoT when energy saving methods are used to save power. In this chapter, we elaborate on these problems and propose a SE-based solution to solve it effectively.

4.2 Certification of Data Affiliation in IoT

With reference to Section 3.4, in most of the real-world implementations, RID cannot directly connect to the blockchain and depends on a trusted intermediary for information retrieval from the blockchain. This is a frequent scenario

in BaaS. On reception, the RID has no means to verify if data came from blockchain. As data is added to the blockchain only after consensus (collective decision), any data retrieved from blockchain signifies that indeed, all participants had a consensus agreeing on that particular data. For example, in the use case of blockchain-based remote firmware update, a manufacturer's signed firmware stored on blockchain is more secured and trusted compared to another signed firmware from the same manufacturer as firmware on blockchain was added only after consensus from the concerned parties (software developers, security watchdogs, etc.).

Hence, it is very crucial to not only ensure data origin (manufacturer in this case) but also data affiliation (i.e. it came from the blockchain). As RID cannot directly connect to blockchain to verify on its own, due to severe overhead constraints, an offline technique for verifying data affiliation is needed for realizing complete data security in IoT. In Section 4.2.1, we address this niche problem by proposing a completely new algorithm called Secure Element based Offline Verification Algorithm (SEOVA) [Deshpande *et al.* 2019a] to verify if the data received through intermediary really belongs to the blockchain.

4.2.1 The SEOVA

SEOVA is a novel algorithm to test blockchain data affiliation i.e. to determine if the data coming from blockchain through an intermediary (like in case of BaaS, RIDs), really belongs to the blockchain. In most blockchain platforms, the block validator, while creating a new block, signs it with its private key. This signed block is then disseminated across the blockchain network and added to the blockchain after consensus.

In a typical blockchain data verification process, to verify if a particular data comes from the blockchain, the block validator's signature in the block containing the particular data are verified. The verification is successful if the signature is verified using the public key from the list of registered validators. However, this approach has several flaws:

- Need to maintain a list of public keys of all the validators in the blockchain network.
- Need to update this list after every new block formation.
- Need to connect to the blockchain to execute the above 2 actions.
- Offline verification hence not possible.
- Critical overhead ramifications for RIDs that are resource-constrained and connected to blockchain via an intermediary.

- Requires trust in the intermediary.

To solve these drawbacks, within SEOVA, we propose to alter this commonly used single signature process in favor of a novel double signature block formation process in conjunction with the apt use of SE to implement the same. A generic physical architecture for implementing SEOVA is given in Figure 4.1. Each validator (node) in the blockchain network has a dedicated SE [MAOSCO Limited 2017].

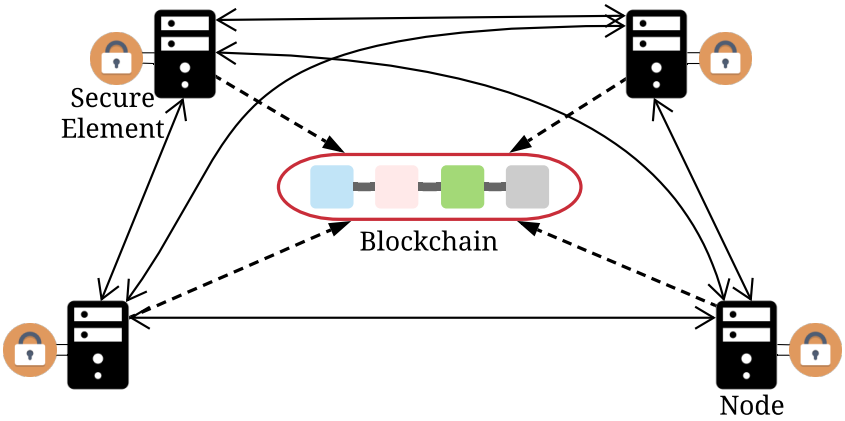


Figure 4.1: Generic physical architecture for SEOVA.

Each of the dedicated SE is instantiated by a trusted entity using a secure process elaborated in [MAOSCO Limited 2017]. During the instantiation, 2 private keys (validator key P_V, blockchain key P_B) and supporting codes are securely injected into the SE. P_V is the unique private key for the particular validator and P_B is the shared private key common to all validators of the blockchain. The corresponding public key (P_B^{pub}) of P_B is injected into the SE of the concerned RIDs (see Fig 3.14).

The process for SEOVA starts at the validator's end. When new data D is to be added to the blockchain, the validator creates a new block B containing this data. This block is then signed by its dedicated SE twice (i.e. with P_V and P_B). This double signed block B_S^2 is then disseminated across all the participating nodes, verified and finally added to the blockchain after consensus. Algorithm 4 illustrates the process in detail which is followed by each node (validator) whenever they want to add new data to blockchain.

Referring to the Scenario 2 (Figure 3.14), whenever, RID receives data (encapsulated in a block) from the blockchain via a trusted/non-trusted intermediary, it simply verifies the signature using P_B^{pub} key to ascertain blockchain affiliation of data. This approach has a wide range of advantages:

- No need for a trusted intermediary.

Algorithm 4 Block formation for SEOVA implementation.

Input: new data D for insertion in blockchain;
Output: double signed block B_S^2 containing new data;

Initialization
get the new data D;
verify data origin $v \leftarrow ECDSA\_verify(D)$;
if $v == failed$ **then**
    $echo$("Data origin cannot be verified"); $exit$;
Block Formation
create new block B with D;
create new block header H_B for B;
insert previous block's $HASH(H_{B-1})$ in H_B;
insert $merkel\_tree$, $timestamp$ in H_B;
Double Signing Using SE
sign $S_V \leftarrow ECDSA\_sign(H_B)$ with P_V;
sign $S_B \leftarrow ECDSA\_sign(H_B)$ with P_B;
Block Finalization
insert S_V, S_B in H_B;
$B_S^2 \leftarrow append(H_B, B)$;
return B_S^2;

- No need to connect to blockchain for verification.
- No overhead on RID as calculation-intensive tasks delegated to SE.
- Validator cannot leak or copy P_B given SE's secure by design architecture.
- No need to change key when validators maintaining blockchain change their individual P_V/new validators are added/removed.
- SE can be programmed to detect hardware tampering attempts and will subsequently cease to function if tampered.
- Identity theft of validators through leaked P_V-P_B, is effectively prevented as keys inside SE cannot be replicated.

4.2.2 Overhead Analysis

In SEOVA, there is an additional usage of SE at blockchain validator level. Since SE is installed as an additional component, we perform overhead analysis for blockchain validator.

4.2.2.1 Computational Overhead

At the blockchain validator level, the delegation of resource-intensive processes like encryption, decryption, signature and its verification to SE significantly reduces the computational overhead on the validator node.

4.2.2.2 Timing Overhead

To measure the timing overhead, we set up a testbed consisting of a blockchain node, and a SE. For SE, we used Multos M5-P19 [Multos form]. For the blockchain node, we used Dell XPS with an Intel i7-8550U processor. To quantize the timing overhead, we performed 100 iterations of ECDSA signature and verification on node and on the SE. The averaged results with standard deviation are presented next.

When compared to blockchain node validator, SE was slower by about 70% for signature (Figure 4.2) and 60% for verification (Figure 4.3. However, SE with its TEE and TSE, was deterministic with negligible standard deviation. The increased time overhead, even though significant in relative terms, accounts only for a few hundred *ms* and gives unmatched security advantages offered by the SE.

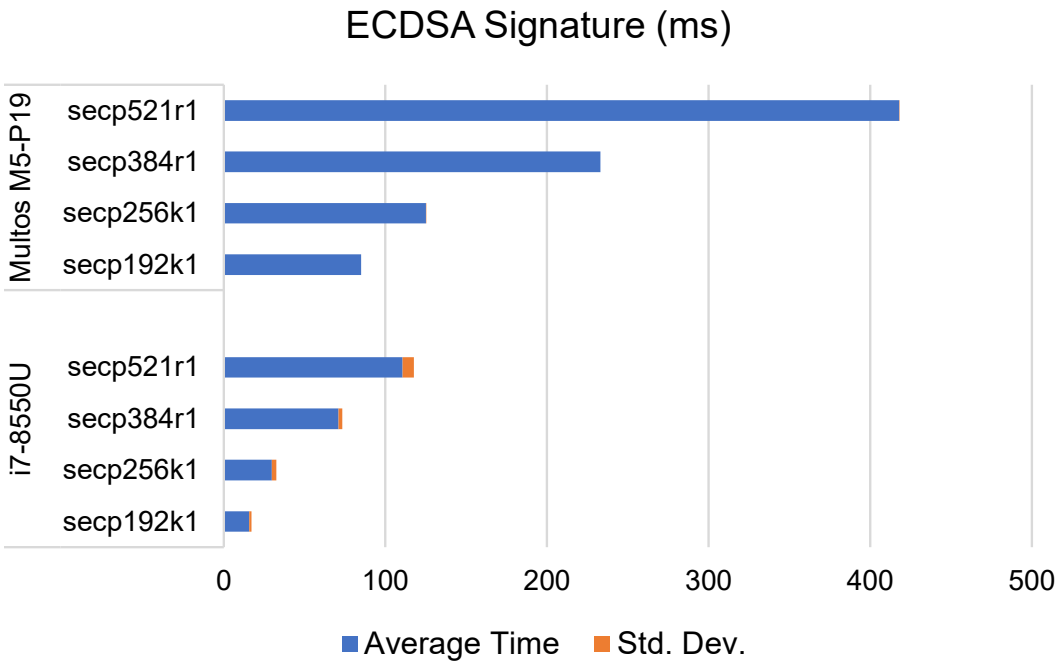


Figure 4.2: ECDSA signature time cycle, blockchain node vs SE.

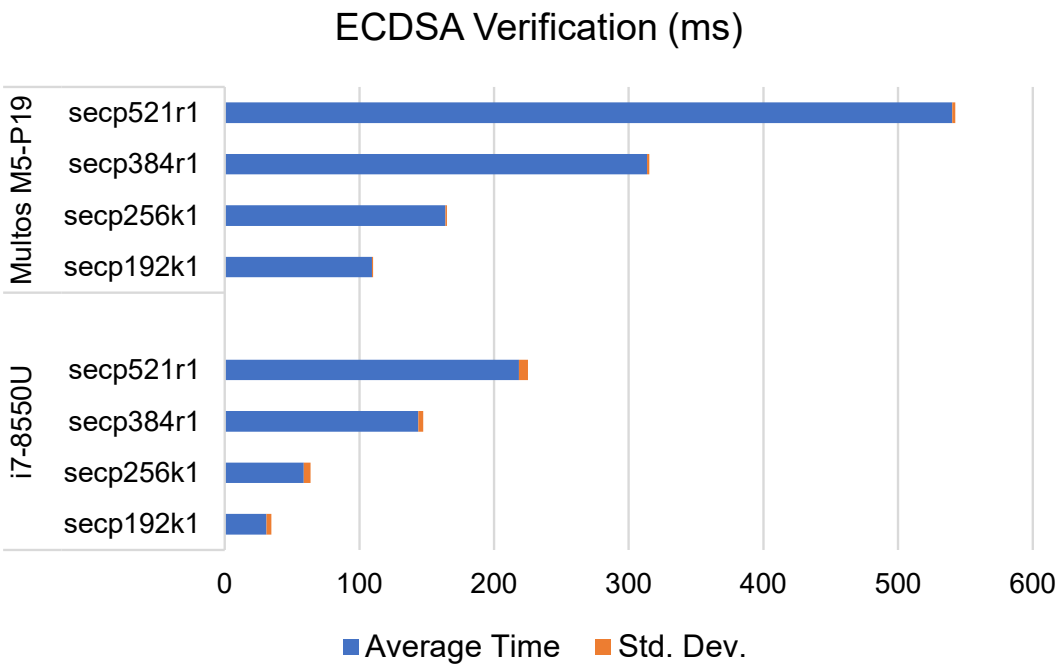


Figure 4.3: ECDSA verification time cycle, blockchain node vs SE.

4.2.2.3 Memory Overhead

Assuming we use ECC [Johnson *et al.* 2001] with a 256-bit curve, for blockchain validator, one public key of 64 bytes for verifying data source and 2 private keys of 32 bytes each, are needed for double signature. As these keys are stored in SE (TSE), therefore, the memory overhead is reduced by 128 bytes. Although with a small factor, the SE reduces overall memory overhead.

4.2.2.4 Cost Overhead

Since additional hardware component installation is required at the blockchain node validator level, the cost overhead is increased. Since, in our proposition of SEOVA, one SE is needed, the cost overhead is between 0.5€ to 10€ depending on the required certification level. In a nutshell, our proposed SEOVA reduce computational, and memory overheads greatly in a given IoT system. Further, when gauged against the benefits, they clearly outweighs the little increased timing and cost overhead.

4.3 Certification of System Liveness in IoT

Liveness is a property that signifies if the system is progressing and free from deadlocks. In our context, this property helps to determine if all the system

end-system(s)/sensor(s) are up and functional.

In the context of IoT, this property is important so as to monitor RID. More so, it is also equally important to check the liveness of connected sub-component(s)/sensor(s) to the RID. However, if the number of sensors connected to any given RID increase, this task can become quickly complex. To elaborate the problem, let us consider the domain of Industry 4.0. In the context of IoT, the Industry 4.0 revolution makes it possible to collect data from thousands of sensors using just a few aggregators. These few hundred aggregators are connected to a centralized aggregator which collects all the data and send it to the cloud for storage, processing and decision making. Because there are potentially numerous sensors, each running with its own periodicity for reporting entity values, it is quite an arduous task to monitor each of them from the cloud.

Industry 4.0 sites generally use Asynchronous Mode [Neumann 2007] for transferring data. This helps the system to be resilient against links failures and insufficient bandwidth errors. Further, to save on processing power, energy and bandwidth, the centralized aggregator may use common methods like "Limited Transmit" (based on RFC 3042 [Allman *et al.* 2001]) to reduce the total number of data transmits. This reduces the overhead cost of sending per bit. To achieve limited transmits, there are numerous methods. For example, under lazy scheduling techniques [Prabhakar *et al.* 2001], an aggregator can choose to transmit the current entity value received from the sensor(s) only when it differs from the past value or if the absolute difference between them is above the indicated threshold.

Such approaches certainly minimize the number of data transmits and the transmission overhead per bit. However, it makes harder to solve an already difficult problem i.e., monitoring the end-site sensor(s) from the cloud. As the data is non-periodic, the cloud monitoring service has to distinguish between two scenarios viz., 1) Everything is fine and data is not transmitted because the transmission conditions are not met, and 2) Data is not transmitted for that sensor(s) or aggregator because of some anomaly/failure (hardware, connection, storage, etc.).

The distinction between these two scenarios is a difficult task because both lead to the same end result i.e., data not being received. It is important to differentiate between the two scenarios, as well as ensure liveness in scenario 1. The probable solution for this, at first, seems to use machine-learning approach [Flovik 2018] which looks on pseudo-periodicity of previous data to flag an alert. However, as the data reception itself is aperiodic, such solution cannot guarantee the genuineness of alert signals. Further, acting on false positives means spending resources, energy for a non-existent problem. Moreover, depending on the complexity of machine-learning algorithms

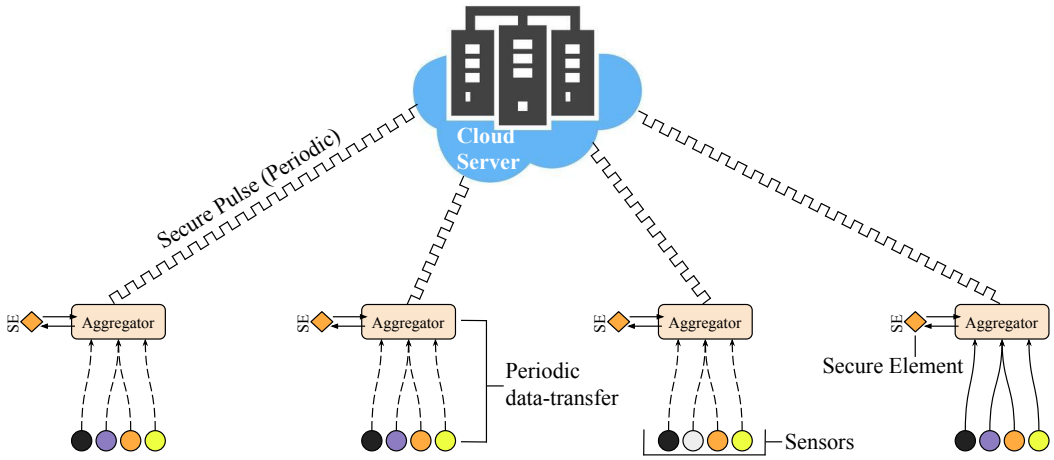


Figure 4.4: PulSec Framework Generic Architecture.

[Williams *et al.* 2006], the processing overhead is high as well. So in our work [Deshpande *et al.* 2019b], we have proposed a framework which not only solves above explained intricate problem but it does that with minimum possible overhead. It also guarantees the genuineness of alerts and goes further to ensure tamper-proofing of the alerts and verify its authenticity. Section 4.3.1 elaborates on the framework of the proposed solution.

4.3.1 PulSec Framework

The inspiration for the solution presented in this section is taken from the mother nature itself. Just as any doctor can certify if a person is dead or alive by checking his/her heart pulse, on the same ground, we present a solution based on a low overhead, secure, digital pulse. Secure pulse in simple terms is a signed payload sent periodically to assess the current state of the sensors and aggregators remotely. This secure pulse (or PulSec, refer Figure 4.4) enables to check the status of not only the device/aggregator but also the sensor(s) attached to it. By automatic monitoring of this secure pulse, a cloud administrator can detect an anomaly in no time and can take corrective measures rapidly. Moreover, with this approach, there is no need to change the data structures of the datasets to be transmitted as it works like a complete add-on with minimum overhead and guaranteed functioning. To realize PulSec, we have used Multos Secure Element M5-P19 [Multos form]. This is used to generate a secure pulse.

We designed our PulSec to remain well within the limit of 1500 Bytes so that packet fragmentation probability and overhead remains very low because, although, the Internet Protocol (IP) supports a packet of size 65536

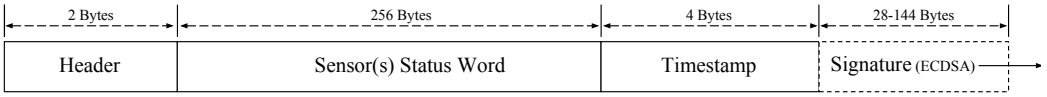


Figure 4.5: Payload construction for secure pulse (Simple PulSec).

Bytes [Postel *et al.* 1981], the Maximum Transferable Unit (MTU) is around 1500 Bytes [Mogul & Deering 1990] for most Data link layers. Hence, anything above 1500 Bytes (including headers) is divided to form smaller packets increasing transmit overhead per bit.

We propose two types of frameworks i.e.,

1. *Simple PulSec*: The *maximum* payload size is 406 Bytes.

2. *Compressed PulSec*: The *minimum* payload size is 34 Bytes.

Both frameworks work on the same principle and the only difference is in the payload construction and size. In simple PulSec (refer Figure 4.5), for construction of this periodic signed pulse, the aggregator runs continuously a small script to check every minute the timestamp of all the last entity values from different sensor and check if the periodicity is satisfied or not[1] with respect to older entity values. Then the status of each sensor is encoded in bit-wise representation. The order of the bit is equal to the pre-established sensors list to make the mapping easy. Bit HIGH is indicated for the sensors satisfying their respective periodicity and LOW for the non-satisfying sensors.

These bits are packed together to form a 256 bytes long packet. The left-most bit indicates the status of the sensor with $id = 1$, the second left-most bit indicates the same for $id = 2$ and so on. As 1 bit per sensor is required for indicating the status, overall $256 * 8 = 2048$ sensors' status can be reported by PulSec in its basic form. This 256 bytes packet is succeeded by the 4 bytes timestamp. This newly formed 260-byte packet is then sent to the secure element for signature. Secure element sign the packet and resend it back to the aggregator along with an appended signature at the end. This signed packet is then preceded by 2-byte header which basically indicates the type of the PulSec (0-3 bit), the type of the signature mechanism used(4-7 bit) and the *uid* of the aggregator (8-15 bit). This 1 byte *uid* helps to segregate the identical PulSec packets arriving concurrently from different remote sites.

Depending on the signature algorithm[2] used, the total payload size varies but is under 500 bytes in all cases. This is then sent to the cloud server.

[1]data is received periodically at the aggregator level and the aperiodic transmission is between aggregator and the cloud server.

[2]we used Elliptic Curve Digital Signature Algorithm (ECDSA) [Johnson *et al.* 2001] because it provides higher level of security for smaller length of keys *compared to Rivest-

In our framework, we propose to send this final packet once every minute i.e., secure pulse with a time period of 60 seconds. This time period allows anomaly/failure detection in $\tau + 60$ seconds[3] (where τ is the periodicity of receiving data for a given sensor). This time period can be increased or decreased considering use-case(s). For compressed PulSec (refer Figure 4.6), the 256 Bytes Sensor(s) Status Word (SSW) is replaced by Sensor Error List (SEL) which contains the list of non-working/failed sensors (if any). This compressed PulSec is beneficial for remote industrial sites where internet bandwidth is not available to a great extent. Moreover, to further reduce the size of the payload, smaller curves like *secp112r1*,*secp112r2* within Elliptic Curve Digital Signature Algorithm (ECDSA) can be used to reduce the size of the signature to 28 Bytes. Calculating further, when there are no errors, the resulting payload becomes 34 bytes (2 bytes header, 4 Bytes timestamp, 28 bytes signature). This method works well till 256/2 = 128 sensors[4] are not working. After this limit, the error list itself becomes longer than SSW (256 bytes) in simple PulSec.

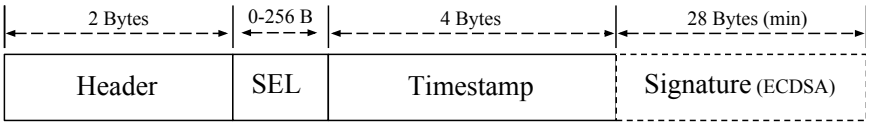


Figure 4.6: Payload for compressed PulSec.

After the pulse is received on the cloud, two processes are carried out viz., Cross-verification and Mapping. When the packet payload is received, according to the header information, the signature is checked against the available public key for given *uid* from the database using the indicated signature algorithm/curve. If the signature verification is successful, then the sensor(s) status is mapped in the database with the corresponding timestamp. In the case of compressed PulSec, the faulty sensor(s) status are updated for further action.

4.3.2 PulSec Algorithms

In this section, the algorithms which are the part of the PulSec framework are illustrated. With reference to section 4.3.1, there are 2 types of PulSec frameworks proposed. Algorithm 5 illustrates simple PulSec generation while Algorithm 6 explains compressed PulSec generation. In Algorithms 5 and

Shamir-Adleman Algorithm (RSA) [Rivest *et al.* 1978], Digital Signature Algorithm (DSA) [Kravitz 1993].

[3] not considering the low magnitude transmission delay ($sensor \rightarrow aggregator \rightarrow cloud$).

[4] 2 bytes for each sensor id representation.

6, the data is sent to secure element for the signature. This signing process within the secure element is outlined in Algorithm 7. Algorithm 8 details on the procedure performed by the cloud server for verification of PulSec for remote device(s)/aggregator(s).

Algorithm 5 Simple PulSec Construction.

$t \leftarrow threshold_{tolerance}$
$SSW[] \leftarrow emptyBitArray$
$S \leftarrow list[getSensorIDs()]$
Require: $2048 \geq size(S)$
for all $sensorID$ in S **do**
  $n \leftarrow index[lastSensorValue]$
  $ts \leftarrow getTimestamp(CURRENT)$
  $ts_n \leftarrow getTimestamp(n)$
  $ts_{n-1} \leftarrow getTimestamp(n-1)$
  if $[ts - ts_n] \leq [ts_n - ts_{n-1}] + t$ **then**
    $SSW[sensorID] = HIGH$
  else
    $SSW[sensorID] = LOW$
$TS \leftarrow getTimestamp(CURRENT)$
$P \leftarrow concatenate(SSW, TS)$
$UART\_SE \leftarrow serialWrite(P)$ {signing by SE}
$P_S \leftarrow serialRead(UART\_SE)$
$H \leftarrow header(Simple, ECDSA_{curve}, uid)$
$PL \leftarrow concatenate(H, P, P_S)$ {payload formation}
Ensure: $ioctl(NIC) \leftarrow sendTCP(PL)$

4.3.3 PulSec Features

At the heart of the PulSec framework lies the secure element. The novel architecture of the secure element which helps to solve many issues of security, tamper-proofing, and future use are explained in this section.

4.3.3.1 Security Level

In terms of security, even when the smallest curves in ECDSA are used (112 bit - secp112r1, secp112r2), the security level they provide is 2^{56} combinations .i.e., an attacker needs to perform 2^{56} combinations within 1 minute (our defined time period) to obtain something meaningful [Girault *et al.* 1988]. To give a perspective on the level of difficulty of this task, one can compare it to

Algorithm 6 Compressed PulSec Construction.

$t \leftarrow threshold_{tolerance}$
$SEL[] \leftarrow emptyIntArray$
$S \leftarrow list[getSensorIDs()]$
$i \leftarrow 0$
for all $sensorID$ in S **do**
  $n \leftarrow index[lastSensorValue]$
  $ts \leftarrow getTimestamp(CURRENT)$
  $ts_n \leftarrow getTimestamp(n)$
  $ts_{n-1} \leftarrow getTimestamp(n-1)$
  if $[ts - ts_n] > [ts_n - ts_{n-1}] + t$ **then**
    $SEL[i] = sensorID$
    $i \leftarrow +1$
$TS \leftarrow getTimestamp(CURRENT)$
$P \leftarrow concatenate(SEL, TS)$
$UART\_SE \leftarrow serialWrite(P)$ {signing by SE}
$P_S \leftarrow serialRead(UART\_SE)$
$H \leftarrow header(Simple, ECDSA_{curve}, uid)$
$PL \leftarrow concatenate(H, P, P_S)$ {payload formation}
Ensure: $ioctl(NIC) \leftarrow sendTCP(PL)$

Algorithm 7 Multos SE Signature Algorithm.

Require: $UP \leftarrow UART\_SE(P)$ {get packet from serial interface}
$hash_{UP} \leftarrow hash(type = SHA256, UP)$
$SIG \leftarrow multosECDSA(hash_{UP}, key, parameters)$ {get signature}
Ensure: $UART\_SE \leftarrow serialWrite(SIG)$ {transmit the signature over serial interface}

Algorithm 8 PulSec Verification.

Require: $receiveTCP(PL) \leftarrow ioctl(NIC)$ {get the PulSec payload}
$H, SSW/SEL, TS, P_S \leftarrow splitPayload(PL)$
$agID \leftarrow getSourceID(H)$
$EC \leftarrow getECCparam(H)$
$PT \leftarrow getPulSecType(H)$
$verfication \leftarrow verifyECDSA(agID, EC, SSW/SEL + TS, P_S)$
if $verfication ==$ **true then**
if $PT == SIMPLE\_PULSEC$ **then**
mapStatus(agID, TS, SSW)
else if $PT == COMPRESSED\_PULSEC$ **then**
mapStatusOFF(agID, TS, SEL) {Map OFF for SEL, ON for the rest}
else
print "Invalid PulSec type" + TS
Ensure: RESTART with new payload
else
print "Invalid Signature" + TS
Ensure: RESTART with new payload

the nonce field in Bitcoin Blockchain. It is a 32-bit field and maximum 2^{32} combinations are required to get the right hash to earn block-reward (12.5 BTC currently) within 10 minutes. In a nutshell, it is much easier (with current processing capabilities) to mine Bitcoin rather than to attack PulSec (2^{24} times more difficult), even when least secure curves are considered.

4.3.3.2 Tamper-Proofing

The signing process in the framework makes sure that the data which is sent is tamper-proof (with current computing capabilities). In our framework, we use the ECDSA to sign the sensor status and timestamp packet. Further, SE provides TEE for the signing process as well as the TSE for storing the private keys. As most modern aggregator(s) in industrial sites lack TEE/TSE, the SE becomes indispensable in this case. Further, the tamper-interrupt pin present on SE is able to detect tamper signal. This helps to further increase the hardware integrity (an important aspect for demand-response). Whenever there are attempts to tamper hardware, the SEs stop working and the cloud server gets notified.

4.3.3.3 Future-Proofing

PulSec is fully prepared for scale-intensive use-case(s) which may arise in future. With the current proposition, it already supports $2^8 = 256$ uids i.e. 256 industrial sites can be surveilled by single cloud-based server. Further, for each uid (or centralized aggregator), $256 * 8 = 2048$ sub-aggregators/sensors are supported. The implementation can easily be changed to increase the size of the header and/or SSW within margin of $1500 - 464 = 1036$ bytes[5] to avoid packet fragmentation at mac-layer. This additional increase would mean an additional monitoring capacity for 8288 sub-aggregators/sensors taking the tally to more than 10k in simple PulSec format. For compressed PulSec, this can go even higher depending on the failure rate of sensors.

4.3.3.4 Application Domains

Apart from Industry 4.0, PulSec can be used anywhere when monitoring is of critical importance. For example, in Energy Demand Response, where monitoring the power consumption response from remote is of utmost importance, there, PulSec could be used to monitor the state of the sensors when critical processes are time-shifted to avoid peaks in consumption. Further, another envisaged example can be in context of autonomous vehicles where PulSec monitors critical sensors (eg. safety, battery, etc.) and alerts the manufacturer which in turn registers the vehicle for the anticipated maintenance.

4.3.4 PulSec Overhead Analysis

As the framework consists of 2 main sites i.e. site of PulSec construction and site for PulSec verification, the overhead can be considered for both sites. However, as the PulSec verification takes place at a cloud-based centralized powerful server, the overheads there are not taken into account considering server has enough bandwidth, processing power, storage capacity and the low magnitude of PulSec overhead does not affect its performance. Also, as SE is considered as an add-on (HSM), the overheads within SE are not considered for simplicity reasons. Further, As PulSec is a monitoring framework, its overhead cannot be directly compared to the tools like Prometheus or protocols like Simple Network Management Protocol (SNMP) as they both where they have both monitoring and management capability. However, appropriate references are given where the comparison is possible.

[5] when biggest supported curves - sect571k1, sect571r1 are considered.

4.3.4.1 Bandwidth and Processing Overhead

On the PulSec construction side (aggregator), for calculating *bandwidth overhead*, we consider the maximum payload from simple PulSec which is 406 bytes. Adding Ethernet Frame (18 bytes), Transport Control Protocol (TCP) header (20 bytes), IPv4 header (20 bytes), the total packet size becomes 464 bytes. This packet is sent every minute in our standard framework proposition. Therefore the bandwidth overhead becomes: $\frac{464}{60} \approx 7.73 bytes/sec.$

For comparison, this is lower than SNMP (25 bytes/sec, [Breit 2003]).Continuing further, for *processing overhead* of the PulSec construction, the most intensive task i.e. ECDSA, considering the processing capabilities of an aggregator, is delegated to secure element and hence the PulSec construction does not cause measurable significant overhead. For verification, ECDSA requires more processing power. However, as this is done on cloud end using servers much more powerful than aggregators, the overhead is negligible.

4.3.4.2 Timing Overhead

For *timing overhead* at PulSec construction point, steps involved are deterministic with deterministic time delays. This is because only 1 cycle of ECDSA is performed per minute. Further, within SE, the hash of the concatenated SSW and Timestamp is signed. As for hashing and signing (on specific hash length), both processes are deterministic, the overhead is constant. During our experimentation, we found that the SE's response time to return the signature was less than $500ms$ even when biggest supported curves (571 bit - sect571k1, sect571r1) where used within ECDSA. This is consistent with other results obtained from tinyECC implementation where response time 460 ms for 160-bit curve [Liu 2007]. The reason why SE is faster is that it has a crypto co-processor to speed up cryptographic processes like the signing, hashing, encryption, decryption, verification. We did not measure timing overhead during a tamper interrupt as our propositional framework stops the PulSec secure pulse once tampering happens and manual resolution is needed.

4.3.4.3 Memory and Cost Overhead

For *memory overhead*, the aggregator needs a maximum of 464 bytes of storage for payload construction which can be reused per minute. For the storage of last active timestamp of different entities, we considered that it is implemented by default in a robust aggregator and hence the overhead for the same is not considered. For comparison, the memory overhead in Prometheus is 5-10 Mb minimum (data from a real deployment).

Further, In PulSec framework, the only hardware component needed per centralized aggregator is secure element. As other overheads are negligible, additional cost overhead does not arise for other components. The cost overhead of SE depends largely on the EAL certification level. Approximate pricing for non-certified SEs is 0.5€ while for EAL7 certified is 10€.

Summary: In this chapter, we addressed the niche problems of data affiliation and sensor monitoring (system liveness) by proposing SE-based SEOVA and PulSec respectively for realizing holistic *transaction* security when our proposed combinational approach is applied. In the next chapter, we conclude by highlighting the contributions of this thesis.

Chapter 5

Conclusion and Perspectives

Contents

5.1 Conclusion

In this thesis, we addressed the impediments faced by traditional decentralized databases for securing transactions or more appropriately atomic data transfers. In Chapter 1, we laid the background of how the introduction of blockchain and its use as a distributed database can solve many impediments as it benefits not only from classical properties of distributed databases such as consensus, guarantee of total order of transactions, reliable multicast, etc. but also adds security and immutability (add-only database).

Blockchain modeling and dimensioning

Following that, in Chapter 2, we addressed the main problems in evaluating blockchain as a suitable replacement for securing transactions. The single most limiting problem for blockchain is the lack of modeling for deriving optimum parameter based on application. Common questions like what is the ideal number of connections per node, how many inbound/outbound connections, optimal diameter value, placement of powerful clusters, etc. were mathematically addressed and validated using simulations. We proposed a mathematical model based on graph theory to derive the optimum number of connections per node required for connectivity and its relation to diameter and hence the block time estimation. We also proposed a topology mapping mechanism to evaluate suitability of already developed blockchain frameworks.

Blockchain with Root of Trust

Next, in Chapter 3, we highlighted that why blockchain alone cannot be used to holistically secure transaction as it guarantees data immutability only whereas in real world scenarios, the data has also to be secured at the point of generation and usage. If data is changed before being stored on a blockchain, it can effectively affect consensus decision as there will be a disagreement between validators. Moreover, given the high overhead and redundancy, we are also aware that blockchain cannot penetrate to lower levels of any system hierarchy essentially limiting its reach. To mitigate this, we proposed the use of SE to establish the *root of trust* since the SE architecture follows the "*secure by design*" paradigm.

Equipped with these two technologies as the base of our proposed decentralized system, in Chapter 3 we applied it to fully secure *transactions* in three disparate fields of Smart Grids, Smart Vehicles and IoT.

Democratizing Demand Response

In Smart Grids, we addressed the problem of designing of a distributed marketplace using the concepts of blockchain, SE, smart contracts, escrow accounts. We addressed the issue of large data storage on blockchain for DR and centralization in DR allotment by successfully designing a decentralized autonomous bidding system.

Secure Key Handling with Responsibility

Next, we applied our system to smart vehicles and addressed the problem of Trusted Execution and Storage Environment within the smart vehicle. With the use of blockchain and SE, we showed how *non-repudiable responsibility* can be enforced. This basically ensures that when regulators audit the data, its veracity can not be denied on technical factors whatsoever. The issue of secure firmware update and secure key management is also addressed and solved.

Holistic Data Security in IoT

With the application of our combinational system to IoT, we solved the important issue of holistic data security in resource constrained devices i.e., securing data at all 3 points viz., point of generation, point of storage, point of usage in an IoT system all while maintaining very low overhead and improving performance by as much as 31 times.

Offline Verification of Blockchain Data

In Chapter 4, We addressed the pressing issue of verification of blockchain data where a remote device which receives data from blockchain through an intermediary and does not have resources and online connectivity to verify it. With the proposition of SEOVA's double signature algorithm using the technology of SE, we successfully solved this issue without compromising on security and privacy. The end device in this case, neither needs online connectivity nor needs additional resources to perform a secure and tamper-proof offline verification.

Efficient Sensor Monitoring

In Chapter 4, Going one step further, we proposed a solution to the classical problem of sensor monitoring where because of emphasis on energy savings, the data from remote sites (like in industry 4.0) is transmitted using "limited transmit" method which results in aperiodicity in transmit patterns for various onsite sensors.

The system on the other end which is connected through the internet, in this case, has to distinguish between two scenarios i.e. 1) data is not transmitted because transmission conditions are not met , and 2) data is not transmitted because of some error on remote site with that particular sensor. With the same outcome for both cases, this is a very arduous task. However, with our proposed PulSec framework based on SE, we not only solved this problem while maintaining very low overhead (bandwidth = 7.73 B/s, memory = 464 B, time = 500 ms) but also we eliminated all false-positive cases.

5.2 Perspectives

This thesis' contributions solve several industrial and real world application problems and we believe that with the approach presented in this thesis, even more problems across many disparate fields can be solved while many of the already implemented solutions could be further improved. We suggest few as follows:

- Short Term:
  - Characterize the energy consumption of blockchains w.r.t. consensus strategies. This approach can further help to change consensus on-demand w.r.t. network activity. For e.g. in healthcare, during a health emergency, the medical transactions on the blockchain can

be confirmed using lighter consensus instead of PoW-based consensus.

- Comparative analysis on usage of SE versus Virtual SE for blockchain applications. The Virtual SE is a secure sandbox [Brudnicki *et al.* 2012] that is embedded within the OS/application which creates a safe operating environment, similar to SE.

- Medium Term: Application of Blockchain + SE approach to other domains -
  - In health monitoring e.g., for elderly people at home, against fraudulent insurance declarations, etc. Since the data source is secured with SE and the subsequent data storage is secured with blockchain, the authenticity of data cannot be denied and thereby creating a reliable trigger to act in emergency situations. This can be even well utilized to tackle fraudulent insurance claims.
  - In the field of smart metering to authenticate and store secured information. This can effectively prevent the tampering attempts of not only the smart meter but also the data from it.

- Long Term:
  - Configuring the blockchain operations like consensus algorithm, P2P network connectivity, block size, block confirmation time, etc., dynamically depending on the application requirements. In the current state of existing blockchains, parameters are fixed. However, configuring some operations like consensus on the go is a much more complicated task given that some consensus are completely different in their functioning. Hence, to implement it, choosing right and compatible pairs is a must.
  - Introduce quality of service mechanisms to arbitrate between important and less important transactions. Using this, we can guarantee any given blockchain's ability to dependably confirm high-priority transactions even during high network activity period. It is an interesting approach, independent from transaction cost. For e.g., it could be used to reliably change on-demand parameters of blockchain so as to quickly adapt it w.r.t. changing scenario.
  - Introduce a mixed criticality scheduling approach [Burns & Davis 2017] for miners having low critical and high critical transactions to mine. High critical transactions can be used for maintenance of the blockchain that should always be

mined in a short time. Mixed criticality scheduling approach has been introduced in the context of safety critical systems to better utilize scheduling resources while always satisfying highly critical tasks at the price of degrading (even pruning) low critical ones. We think that this approach could be interesting to mitigate cyber-physical attacks. For e.g., in the case of a DDoS attack, the system load will increase and may become overloaded. The focus during such scenario is shifting to highly critical transactions validating important information to preserve the consistency of the blockchain.

The perspectives for **long term** can be implemented in two ways i.e., distributed or centralized. While the distributed mechanism can consist of developing a smart contracts-based system for altering the blockchain configurations, the centralized mechanism can apply the SDN paradigm. The tricky part will be to appropriately calibrate the trigger action to activate/initiate the updated blockchain parameters.

Author's Publications

Journal Papers

[*Under Revision*] Varun Deshpande, Hakim Badis, and Laurent George. "Efficient Topology Control of Blockchain Peer to Peer Network Based on SDN Paradigm." In *Peer-to-Peer Networking and Applications*, Springer.

Varun Deshpande, Laurent George, and Hakim Badis. "PulSec: Secure Element Based Framework for Sensors Anomaly Detection in Industry 4.0." In *IFAC PapersOnLine 52.13 (MIM 2019)*, Volume 52, Issue 13, DOI: 10.1016/j.ifacol.2019.11.362. Elsevier, 2019.

Conference Papers

Varun Deshpande, Laurent George, Hakim Badis, and Alemayehu Addisu Desta. "Blockchain Based Decentralized Framework for Energy Demand Response Marketplace." In *IEEE/IFIP Network Operations and Management Symposium (NOMS 2020)*, Budapest, Hungary, pages 1-9, DOI: 10.1109/NOMS47738.2020.9110378. IEEE, 2020.

Varun Deshpande, Taniya Das, Hakim Badis, and Laurent George. "SEBS: A Secure Element and Blockchain Stratagem for Securing IoT." In *IEEE Global Information Infrastructure and Networking Symposium (GIIS 2019)*, Paris, France, pages 1-7, DOI: 10.1109/GIIS48668.2019.9044957. IEEE, 2019.

Varun Deshpande, Laurent George, and Hakim Badis. "SaFe: A Blockchain and Secure Element Based Framework for Safeguarding Smart Vehicles." In *12th IFIP Wireless and Mobile Networking Conference (WMNC 2019)*, Paris, France, pages 181-188, DOI: 10.23919/WMNC.2019.8881408. IEEE, 2019.

Varun Deshpande, Hakim Badis, and Laurent George. "BTCmap: Mapping Bitcoin Peer-to-Peer Network Topology." In *IFIP/IEEE International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks (PEMWN 2018)*, Toulouse, France, 2018, pages 1-6, DOI: 10.23919/PEMWN.2018.8548904. IEEE, 2018.

Bibliography

[Adams & Lloyd 1999] Carlisle Adams and Steve Lloyd. Understanding public-key infrastructure: concepts, standards, and deployment considerations. Sams Publishing, 1999. (Cited on page 63.)

[Aircraft Accident Investigation Bureau 2019] Aircraft Accident Investigation Bureau. *Aircraft Accident Investigation Bureau Preliminary Report*, March, 2019. (Cited on page 75.)

[Allman *et al.* 2001] M Allman, H Balakrishnan and S Floyd. *RFC 3042: Enhancing TCP's loss recovery using limited transmit*. January2001, 2001. (Cited on page 97.)

[Baldoni *et al.* 2000] Roberto Baldoni, J-M Helary and Michel Raynal. *From crash fault-tolerance to arbitrary-fault tolerance: Towards a modular approach*. In Proceeding International Conference on Dependable Systems and Networks. DSN 2000, pages 273–282. IEEE, 2000. (Cited on page 9.)

[Banton 2019] Caroline Banton. *Escrow: What is Escrow?*, 2019. (Cited on page 61.)

[Barzdin 1973] Ia M Barzdin. Finite automata-behavior and synthesis. North-Holland Publishing Company, 1973. (Cited on page 35.)

[Belova & Ouyang 2017] M. Belova and M. Ouyang. *Breadth-First Search with A Multi-Core Computer*. In Parallel and Distributed Processing Symposium Workshops (IPDPSW), 2017 IEEE International, pages 579–587, 2017. (Cited on page 53.)

[Beverly & Afergan 2007] Robert Beverly and Mike Afergan. *Machine learning for efficient neighbor selection in unstructured p2p networks*. SysML, vol. 7, pages 1–6, 2007. (Cited on page 20.)

[Biryukov *et al.* 2014] Alex Biryukov, Dmitry Khovratovich and Ivan Pustogarov. *Deanonymisation of clients in Bitcoin P2P network*. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, pages 15–29. ACM, 2014. (Cited on page 54.)

[bitcoin.it] bitcoin.it. *Satoshi Client Node Discovery*. (Cited on page 18.)

[Breit 2003] Lou Breit. *SNMP overhead and performance impact*, 2003. (Cited on page 105.)

[Brudnicki *et al.* 2012] David Brudnicki, Michael Craft, Hans Reisgies and Andrew Weinstein. *System and Method for Providing a Virtual Secure Element on a Portable Communication Device*, May 2012. US Patent App. 13/279,147. (Cited on page 110.)

[Burns & Davis 2017] Alan Burns and Robert I. Davis. *A Survey of Research into Mixed Criticality Systems*. ACM Comput. Surv., vol. 50, no. 6, November 2017. (Cited on page 110.)

[Cachin 2016] Christian Cachin. *Architecture of the hyperledger blockchain fabric*. In Workshop on distributed cryptocurrencies and consensus ledgers, volume 310, page 4, 2016. (Cited on page 57.)

[Castro & Liskov 1999] Miguel Castro and Barbara Liskov. *Practical Byzantine Fault Tolerance*. In Proceedings of the Third Symposium on Operating Systems Design and Implementation, OSDI '99, pages 173–186, USA, 1999. USENIX Association. (Cited on page 9.)

[Chen *et al.* 2010] Lijun Chen, Na Li, Steven H Low and John C Doyle. *Two market models for demand response in power networks*. In 2010 First IEEE International Conference on Smart Grid Communications, pages 397–402. IEEE, 2010. (Cited on page 66.)

[Chen *et al.* 2017] Lin Chen, Lei Xu, Nolan Shah, Zhimin Gao, Yang Lu and Weidong Shi. *On security analysis of proof-of-elapsed-time (poet)*. In International Symposium on Stabilization, Safety, and Security of Distributed Systems, pages 282–297. Springer, 2017. (Cited on page 9.)

[Chen 2000] Zhiqun Chen. Java card technology for smart cards: architecture and programmer's guide. Addison-Wesley Professional, 2000. (Cited on page 10.)

[CoinMarketCap 2018] CoinMarketCap. *All Cryptocurrencies*, December 2018. Online. (Cited on page 4.)

[Conejo *et al.* 2010] Antonio J Conejo, Juan M Morales and Luis Baringo. *Real-time demand response model*. IEEE Transactions on Smart Grid, vol. 1, no. 3, pages 236–242, 2010. (Cited on page 66.)

[Dannen 2017] Chris Dannen. Introducing ethereum and solidity. Springer, 2017. (Cited on page 57.)

[de Carvalho Silva *et al.* 2017] J. de Carvalho Silva, J. J. P. C. Rodrigues, A. M. Alberti, P. Solic and A. L. L. Aquino. *LoRaWAN - A low power WAN protocol for Internet of Things: A review and opportunities*. In 2017 2nd International Multidisciplinary Conference on Computer and Energy Science (SpliTech), pages 1–6, 2017. (Cited on page 6.)

[Deshpande *et al.* 2018] Varun Deshpande, Hakim Badis and Laurent George. *BTCmap: Mapping Bitcoin Peer-to-Peer Network Topology*. In 2018 IFIP/IEEE International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks (PEMWN), pages 1–6. IEEE, 2018. (Cited on pages 4, 37, 39 and 48.)

[Deshpande *et al.* 2019a] Varun Deshpande, Taniya Das, Hakim Badis and Laurent George. *Sebs: A secure element and blockchain stratagem for securing iot*. In 2019 Global Information Infrastructure and Networking Symposium (GIIS), pages 1–7. IEEE, 2019. (Cited on pages 10, 20, 83 and 92.)

[Deshpande *et al.* 2019b] Varun Deshpande, Laurent George and Hakim Badis. *Pulsec: Secure element based framework for sensors anomaly detection in industry 4.0*. IFAC-PapersOnLine, vol. 52, no. 13, pages 1204–1209, 2019. (Cited on pages 10, 20 and 98.)

[Deshpande *et al.* 2019c] Varun Deshpande, Laurent George and Hakim Badis. *Safe: A blockchain and secure element based framework for safeguarding smart vehicles*. In 2019 12th IFIP Wireless and Mobile Networking Conference (WMNC), pages 181–188. IEEE, 2019. (Cited on pages 4, 20 and 73.)

[Deshpande *et al.* 2020] Varun Deshpande, Laurent George, Hakim Badis and Alemayehu Addisu Desta. *Blockchain Based Decentralized Framework for Energy Demand Response Marketplace*. In NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium, pages 1–9. IEEE, 2020. (Cited on pages 20 and 56.)

[Diot *et al.* 1997] Christophe Diot, Walid Dabbous and Jon Crowcroft. *Multipoint communication: a survey of protocols, functions, and mechanisms*. IEEE journal on selected areas in communications, vol. 15, no. 3, pages 277–290, 1997. (Cited on page 58.)

[Documentation 2000] Python Support Documentation. *Python Software Foundation*, 2000. (Cited on page 50.)

[Dwork *et al.* 1988] Cynthia Dwork, Nancy Lynch and Larry Stockmeyer. *Consensus in the Presence of Partial Synchrony.* J. ACM, vol. 35, no. 2, pages 288–323, April 1988. (Cited on page 9.)

[FIPS 2012] PUB FIPS. *180-4.* Secure hash standard (SHS)," March, 2012. (Cited on page 81.)

[Fischer *et al.* 1985] Michael J. Fischer, Nancy A. Lynch and Michael S. Paterson. *Impossibility of Distributed Consensus with One Faulty Process.* J. ACM, vol. 32, no. 2, pages 374–382, April 1985. (Cited on page 9.)

[Flaxman & Frieze 2004] Abraham D Flaxman and Alan M Frieze. *The diameter of randomly perturbed digraphs and some applications.* In Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, pages 345–356. Springer, 2004. (Cited on page 35.)

[Flovik 2018] Vegard Flovik. *How to use machine learning for anomaly detection and condition monitoring.* https://towardsdatascience.com/how-to-use-machine-learning-for-anomaly-detection-and-condition-monitoring-6742f82900d7, 2018. (Cited on page 97.)

[Floyd *et al.* 1997] Sally Floyd, Van Jacobson, C-G Liu, Steven McCanne and Lixia Zhang. *A reliable multicast framework for light-weight sessions and application level framing.* IEEE/ACM transactions on networking, no. 6, pages 784–803, 1997. (Cited on page 58.)

[Frieze & Karoński 2016] Alan Frieze and Michał Karoński. Introduction to random graphs. Cambridge University Press, 2016. (Cited on page 30.)

[Gervais *et al.* 2015] Arthur Gervais, Hubert Ritzdorf, Ghassan O. Karame and Srdjan Capkun. *Tampering with the Delivery of Blocks and Transactions in Bitcoin.* In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, CCS '15, page 692–705, New York, NY, USA, 2015. Association for Computing Machinery. (Cited on page 9.)

[Girault *et al.* 1988] Marc Girault, Robert Cohen and Mireille Campana. *A generalized birthday attack.* In Workshop on the Theory and Application of of Cryptographic Techniques, pages 129–156. Springer, 1988. (Cited on page 101.)

[GlobalPlatform, Inc. 2018] GlobalPlatform, Inc. *Introduction to Secure Element*, 2018. (Cited on pages 10 and 20.)

[Greenspan 2015] Gideon Greenspan. *MultiChain private blockchain—White paper*. URl: http://www. multichain. com/download/MultiChain-White-Paper. pdf, 2015. (Cited on pages 57, 60, 61 and 76.)

[Haber & Stornetta 1990] Stuart Haber and W Scott Stornetta. *How to time-stamp a digital document*. In Conference on the Theory and Application of Cryptography, pages 437–455. Springer, 1990. (Cited on page 4.)

[Hasse *et al.* 2016] Felix Hasse, Axel von Perfall, Thomas Hillebrand, Erwin Smole, Lena Lay and Maximilian Charlet. *Blockchain–an opportunity for energy producers and consumers*. PwC Global Power & Utilities, pages 1–45, 2016. (Cited on page 57.)

[IEEE Instrumentation and Measurement Society 2008] IEEE Instrumentation and Measurement Society. *IEEE Std1588TM–2008*, 2008. (Cited on page 65.)

[Johnson *et al.* 2001] Don Johnson, Alfred Menezes and Scott Vanstone. *The elliptic curve digital signature algorithm (ECDSA)*. International journal of information security, vol. 1, no. 1, pages 36–63, 2001. (Cited on pages 76, 88, 96 and 99.)

[Jurgensen & Guthery 2002] Timothy M Jurgensen and Scott B Guthery. Smart cards: the developer's toolkit. Prentice Hall Professional, 2002. (Cited on page 10.)

[Karantias *et al.* 2020] Kostis Karantias, Aggelos Kiayias and Dionysis Zindros. *Proof-of-burn*. In International Conference on Financial Cryptography and Data Security, pages 523–540. Springer, 2020. (Cited on page 9.)

[Karp 1972] R. Karp. *Reducibility among combinatorial problems*. In R. Miller and J. Thatcher, editors, Complexity of Computer Computations, pages 85–103. Plenum Press, 1972. (Cited on page 67.)

[Kocher *et al.* 2018] Paul Kocher, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz and Yuval Yarom. *Spectre attacks: Exploiting speculative execution*. arXiv preprint arXiv:1801.01203, 2018. (Cited on page 74.)

[Kravitz 1993] David W Kravitz. *Digital signature algorithm*, July 27 1993. US Patent 5,231,668. (Cited on pages 77 and 100.)

[Lamport *et al.* 1982] Leslie Lamport, Robert Shostak and Marshall Pease. *The Byzantine Generals Problem*. ACM Trans. Program. Lang. Syst., vol. 4, no. 3, pages 382–401, July 1982. (Cited on page 8.)

[Land & Doig 1960] A. H. Land and A. G. Doig. *An Automatic Method of Solving Discrete Programming Problems*. Econometrica, vol. 28, no. 3, pages 497–520, 1960. (Cited on page 68.)

[Lavric *et al.* 2019] A. Lavric, A. I. Petrariu and V. Popa. *Long Range SigFox Communication Protocol Scalability Analysis Under Large-Scale, High-Density Conditions*. IEEE Access, vol. 7, pages 35816–35825, 2019. (Cited on page 6.)

[Lipp *et al.* 2018] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Anders Fogh, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin*et al. Meltdown: Reading kernel memory from user space*. In 27th {USENIX} Security Symposium ({USENIX} Security 18), pages 973–990, 2018. (Cited on page 74.)

[Liu 2007] An Liu. *TinyECC: Elliptic curve cryptography for sensor networks (version 1.0)*. http://discovery. csc. ncsu. edu/software/TinyECC/, 2007. (Cited on page 105.)

[LocalEthereum 2017] LocalEthereum, 2017. (Cited on page 61.)

[MAOSCO Limited 2017] MAOSCO Limited. *Guide to Loading and Deleting*. MAO-DOC-TEC-008 v2.28, Page 5-22, 2017. (Cited on pages 75 and 93.)

[Mauldin 1982] R. Daniel Mauldin, editor. The scottish book: Mathematics from the scottish cafe. Birkhause, 1 édition, April 1982. (Cited on pages 25 and 33.)

[McGrew *et al.* 2011] D McGrew, K Igoe and Margaret Salter. *Fundamental elliptic curve cryptography algorithms*. Technical report, Internet Engineering Task Force, 2011. (Cited on page 77.)

[Mehar *et al.* 2019] Muhammad Izhar Mehar, Charles Louis Shier, Alana Giambattista, Elgar Gong, Gabrielle Fletcher, Ryan Sanayhie, Henry M Kim and Marek Laskowski. *Understanding a revolutionary and flawed grand experiment in blockchain: the DAO attack*. Journal of Cases on Information Technology (JCIT), vol. 21, no. 1, pages 19–32, 2019. (Cited on page 61.)

[Miller *et al.* 2015] Andrew Miller, James Litton, Andrew Pachulski, Neal Gupta, Dave Levin, Neil Spring and Bobby Bhattacharjee. *Discovering bitcoin's public topology and influential nodes*. et al, 2015. (Cited on page 54.)

[Mogul & Deering 1990] Jeffrey C Mogul and Steven E Deering. *Path MTU discovery*, 1990. (Cited on page 99.)

[Monax] Monax. *Hyperledger Burrow*. (Cited on page 57.)

[Multos Consortium 2020] Multos Consortium. *Multos*, 2020. (Cited on page 10.)

[Multos form] Multos. *ML5-P19 and MC5-P19 on Infineon SLE78 Platform*. https://www.multos.com/products/approved_platforms/MIR/multos_international/m5_p19, SLE78 Platform. (Cited on pages 76, 87, 95 and 98.)

[Nakamoto *et al.* 2008] Satoshi Nakamoto*et al. Bitcoin: A peer-to-peer electronic cash system*, 2008. (Cited on page 4.)

[Neumann 2007] Peter Neumann. *Communication in industrial automation—What is going on?* Control Engineering Practice, vol. 15, no. 11, pages 1332–1347, 2007. (Cited on page 97.)

[Nguyen & Kim 2018] Giang-Truong Nguyen and Kyungbaek Kim. *A Survey about Consensus Algorithms Used in Blockchain.* Journal of Information processing systems, vol. 14, no. 1, 2018. (Cited on page 9.)

[NXP MX6Q] NXP. *RD-IMX6Q-SABRE by NXP Semiconductors*, -IMX6Q. (Cited on page 76.)

[OpenADR Alliance 2013] OpenADR Alliance. *OpenADR 2.0 profile specification*. document 20120912–1, 2013. (Cited on page 68.)

[Pasqualetti *et al.* 2012] F. Pasqualetti, A. Bicchi and F. Bullo. *Consensus Computation in Unreliable Networks: A System Theoretic Approach.* IEEE Transactions on Automatic Control, vol. 57, no. 1, pages 90–104, 2012. (Cited on page 9.)

[Postel *et al.* 1981] Jon Postel*et al. RFC 791: Internet protocol*, 1981. (Cited on page 99.)

[Prabhakar *et al.* 2001] Balaji Prabhakar, E Uysal Biyikoglu and Abbas El Gamal. *Energy-efficient transmission over a wireless link via lazy packet scheduling*. In Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society, pages 386–394. IEEE, 2001. (Cited on page 97.)

[Remote 2014] APDU Remote. *Call Secure (RACS), draft-urien-core-racs-03. txt*. IETF draft August, 2014. (Cited on page 10.)

[Rivest *et al.* 1978] Ronald L Rivest, Adi Shamir and Leonard Adleman. *A method for obtaining digital signatures and public-key cryptosystems*. Communications of the ACM, vol. 21, no. 2, pages 120–126, 1978. (Cited on pages 77 and 100.)

[Srivatsa *et al.* 2006] Mudhakar Srivatsa, Bugra Gedik and Ling Liu. *Large scaling unstructured peer-to-peer networks with heterogeneity-aware topology and routing*. IEEE Transactions on Parallel and Distributed Systems, vol. 17, no. 11, pages 1277–1293, 2006. (Cited on page 20.)

[Urien 2015] Pascal Urien. *RACS: Remote APDU call secure creating trust for the internet*. In Collaboration Technologies and Systems (CTS), 2015 International Conference on, pages 351–357. IEEE, 2015. (Cited on page 10.)

[Urien 2018] Pascal Urien. *Towards secure elements for trusted transactions in blockchain and blochchain IoT (BIoT) Platforms. Invited paper*. In 2018 Fourth International Conference on Mobile and Secure Services (MobiSecServ), pages 1–5. IEEE, 2018. (Cited on pages 20 and 78.)

[Van Bulck *et al.* 2018] Jo Van Bulck, Marina Minkin, Ofir Weisse, Daniel Genkin, Baris Kasikci, Frank Piessens, Mark Silberstein, Thomas F Wenisch, Yuval Yarom and Raoul Strackx. *Foreshadow: Extracting the keys to the intel {SGX} kingdom with transient out-of-order execution*. In 27th {USENIX} Security Symposium ({USENIX} Security 18), pages 991–1008, 2018. (Cited on page 74.)

[Wang *et al.* 2019] W. Wang, D. T. Hoang, P. Hu, Z. Xiong, D. Niyato, P. Wang, Y. Wen and D. I. Kim. *A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks*. IEEE Access, vol. 7, pages 22328–22370, 2019. (Cited on pages 8 and 9.)

[Wilcox 2003] Kenneth R Wilcox. *Multi-application smart cards: Card operating systems and application security.* In 21st Computer Science Seminar, 2003. (Cited on page 11.)

[Williams *et al.* 2006] Nigel Williams, Sebastian Zander and Grenville Armitage. *A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification.* ACM SIGCOMM Computer Communication Review, vol. 36, no. 5, pages 5–16, 2006. (Cited on page 98.)

[Yeow & Lopp 2013] Addy Yeow and Jameson Lopp. *Bitnodes API v1.0*, 2013. (Cited on page 48.)

[Yu & Shao-Hai 2006] Guan Yu and Hu Shao-Hai. *Establishing TCP connections between hosts behind NATs.* In Wireless, Mobile and Multimedia Networks, 2006 IET International Conference on, pages 1–3. IET, 2006. (Cited on page 22.)