BLOCKCHAIN-BASED IOT RESOURCE SHARING WITH PRIVACY PRESERVED TRANSACTIONS

A Thesis

by

Muhammad Yasir

Submitted to the
Graduate School of Sciences and Engineering
In Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in the
Department of Computer Science

Özyeğin University
December 2022

BLOCKCHAIN-BASED IOT RESOURCE SHARING WITH PRIVACY PRESERVED TRANSACTIONS

Approved by:

Assistant Professor Kübra Kalkan Çakmakçı, Advisor
Department of Computer Science
Özyeğin University

Professor Fatih Alagöz
Department of Computer Engineering
Boğaziçi University

Associate Professor Hasan Sözer
Department of Computer Science
Özyeğin University

Date Approved: 20 December 2022

Dedicated to my parents, who instilled extraordinary self-belief and confidence in me since my childhood. I have only made it this far owing to their wonderful support, encouragement & vision. I am extremely fortunate to have been born to them.

ABSTRACT

IoT devices constitute an important component of Industry 4.0 paradigm, but are greatly hindered by their inherent resource constraints. Resource sharing is therefore an essential operating requirement for these devices but lack of privacy and heavy reliance on centralized architectures pose a serious risk for stable functioning. Use of decentralized and high availability platforms like distributed ledgers can provide a divergent and distributed networking conditions but leaves any inter-device interactions completely exposed to third party view. To resolve this, we utilize an innovative combination of smart contracts alongside a zero-knowledge proof generation application, namely Tornado Cash, to align IoT devices on a distributed resource exchange platform with privacy guarantees. In concert with public-key cryptography, our solution provides a framework for resource constrained IoT machines to interact through the blockchain for resource exchange purposes with strong privacy guarantees for both devices. Absolute anonymity is ensured by the protocol's inherent architecture, meaning that conversing devices know almost nothing about each other and consequently no participant or adversary can trivially breach the privacy of either participant. Performance evaluations with a competing & comparatively unsecure protocol yield promising outcomes in terms of blockchain metrics like gas usage & incurred transaction costs

ÖZETÇE

Nesnelerin İnterneti cihazları Endüstri 4.0 paradigmasının önemli bir parçasını oluşturmaktadır, ancak içsel kaynak kısıtlamaları nedeniyle büyük ölçüde engellenmektedir. Bu nedenle kaynak paylaşımı bu cihazlar için temel bir işletim gereksinimidir, ancak gizlilik eksikliği ve merkezi mimarilere aşırı bağımlılık istikrarlı işleyiş için ciddi bir risk oluşturmaktadır. Dağıtık defterler gibi merkezi olmayan ve yüksek kullanılabilirliğe sahip platformların kullanılması farklı ve dağıtık bir ağ koşulları sağlayabilir ancak cihazlar arası etkileşimleri tamamen üçüncü tarafların etkisine açık bırakır. Bunu çözmek için, IoT cihazlarını gizlilik garantileriyle dağıtılmış bir kaynak değişim platformunda hizalamak üzere Tornado Cash adlı sıfır bilgi kanıtı oluşturma uygulamasının yanı sıra akıllı sözleşmelerin yenilikçi bir kombinasyonunu kullanıyoruz. Açık anahtarlı kriptografi ile birlikte çözümümüz, kaynak kısıtlı IoT makinelerinin her iki cihaz için de güçlü gizlilik garantileri ile kaynak değişimi yapabilmelerini blok zinciri üzerinden sağlar. Mutlak anonimlik, protokolün doğal mimarisi ile sağlanır, yani konuşan cihazlar birbirleri hakkında neredeyse hiçbir şey bilmez ve sonuç olarak hiçbir katılımcı veya düşman, herhangi bir katılımcının gizliliğini ihlal edemez. Rakip & nispeten güvensiz bir protokolle yapılan performans değerlendirmeleri, gaz kullanımı & gerçekleşen işlem maliyetleri gibi blok zinciri ölçümleri açısından umut verici sonuçlar vermektedir

ACKNOWLEDGEMENTS

To begin with, I would like to express my absolute gratitude to my thesis supervisor, Prof Dr. Kübra Kalkan, for her invaluable tutelage, support, and encouragement throughout the research and writing process. It was great fun and quite exciting working with her.

Moreover, I am immensely grateful to my committee members Prof Dr. Hasan Sözer and Prof Dr. Fatih Alagöz, for devoting their extremely valuable time, expertise & effort into reviewing my thesis.

I would like to thank all my friends and family who offered an encouraging and positive outlook during the times I was struggling and in doubt. In particular, I am grateful to my parents, who were a constant source of motivation, love and energy throughout my studies.

I would like to express my sincere appreciation and acknowledgements to a few kind strangers from the internet who were instrumental in making this thesis possible. These include Samuel JJ Gosling (aka Gozzy), who maintained the Tornado Cash (TC) software repository, and Micah Zoltu, who provided helpful assistance and feedback with TC. I would also like to give my thanks to Brandon Etter for his help with initially setting up and running TC for the first time.

Conclusively, I would like to acknowledge my roommate, fellow engineer and great friend, Alper Ekmekçioğlu for his upbeat disposition, supportive behavior, insightful advice and for believing in me when it really mattered.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Interdevice communication in the field of computing has been possible since the 1970s [1] and it coincided with the advent of the concept of network sharing [2] in the same time period. This gave rise to interconnected computers and eventually, to internet [3] Accompanied by the advances in technology, it has become possible to reduce the size of a self-contained computing system from that of a large room to a minuscule device the size of an average human palm.

Furthermore, the computing capability has been extended to devices which are not normally considered as computers themselves. This has lead to the invention of IoT devices [4] , which allow seemingly ordinary devices like a refrigerator or a television to have the capability to communicate with other devices and exchange information by digital means. However, IoT devices do not have the processing power of a full-fledged server machine or even that of a laptop computer that is intended for strictly casual or light-weight usage. This brings the concept of resource sharing into the fold.

Resource sharing can be defined as making a resource available on a computer network for another device to access remotely and in a transparent fashion, as if it the resource was local to the recipient device itself. IoT devices can be galvanized through an effective resource sharing mechanism, where other devices can help these devices by lending their computational power. Conversely, the IoT devices themselves can lend their unique capabilities to other IoT devices in the network. For instance, a visual sensor device might require some audio input which it cannot directly arrange for but it can delegate this request to another IoT device in the network. However, there needs to be an underlying security mechanism, that allows this resource sharing process to proceed seamlessly and remain resilient,

even in the face of malicious actors within the network. A Blockchain / Distributed Ledger is deemed suitable for this very purpose, since its inherent and public peer-to-peer architecture can be utilized for anonymous communication between devices while simultaneously minimizing the risk of adversarial conduct from either of the participating devices. Renowned for their immutability, decentralization and data transparency, blockchain ledgers can provide a secure interaction layer which underlies the core application logic. IoT devices operating on top of this layer can securely exchange resources and value without the risk of their privacy being encroached. However, it is non-trivial to manage storing a full node encompassing the entire blockchain, if majority of the network participants happen to be IoT machines. Therefore, a relief mechanism is vital for these resource-constrained devices, so that they can operate seamlessly without being burdened with storing entire blockchain ledgers.

This bring us to the data storage paradigm. It is universally established that, at least for the time being, IoT cannot handle large storage quantities. However, it can act as a light node [5] to simply perform quick read operations for verifying block headers and technically, be a part of the blockchain. However, in the event that the node needs to query the blockchain, it is possible that accessing or reading relevant data from the blockchain can introduce issues of latency and potentially expensive time delays. All this could happen because the IoT device is itself incapable of storing large data quantities. It is, however, possible to use a reliable data solution to help it out. Solutions like the Inter Planetary File System (IPFS) [6] can help in storing comparatively smaller but time-critical data within a peer-to-peer, distributed file-system, the technical anatomy of whom somewhat coincidentally resembles the blockchain itself.

Lastly, there remains the issue of data privacy. We briefly mentioned data transparency but privacy preservation remains a matter that is sensitive and subject to strict legislation [7]. Also, anonymity is a large concern, especially among technological aware organization. Most organizations want to remain anonymous

to the highest degree while doing business online. However, they eventually have to expose sensitive information even to unnecessary third parties. Zero Knowledge proofs [8] offer a fairly strong guarantee of data privacy and ensure that both, data veracity and data integrity, are intact without divulging the actual data.

In this thesis, we utilized a hybrid approach, by combining zero-knowledge proofs with a resource sharing scheme on top of a blockchain, aimed at making the intermingling of resource-constrained devices seamless while secure at the same time. A comprehensive protocol is proposed. It makes use of smart contracts [9] in conjunction with IPFS & ethereum events and is streamlined for resource sharing. Moreover, end to end data integrity is guaranteed courtesy of utilizing asymmetric key cryptography [10]. Even privacy in between negotiating devices is emphasized upon, since the IoT devices remain absolutely anonymous to each other throughout the entire resource sharing process. Save for the public key, the network participant are entirely oblivious of any other information about the device they are dealing with. Instead a trust-less blockchain, with the help of smart contracts matches resource seekers with appropriate resource providers in a private fashion. Accordingly, the protocol is pitted against a counter-part which lacks the zero-knowledge capability and relies on third party. Evaluation outcomes reveal the presented research to be comparatively feasible in addition to offering enhanced security and privacy.

1.1 Motivation

The term **Internet of Things (IoT)** was first coined by Kevin Ashton [4] in 1999. The term "Internet of Things" refers to the vision of a world where all objects are connected and able to communicate and share data. Since then, IoT has evolved and expanded to become a topic of research and significant interest in various fields, including computer science and engineering. Researchers have studied various aspects of the IoT, including its technical and architectural challenges, applications and use cases, as well as the accompanying social and ethical

implications.

Among those, one of the most sensitive and critical issues is that of privacy. Prior studies [11] have taken an analytical look at this particular challenge, but practical solutions remain under developmental phases. Additionally, there's the problem of resource limitation that IoT devices face since they are low-powered [12], mostly small and portable. This naturally makes them susceptible to resource constraints. As a result, IoT devices are unable to harness the computational vigor of a run-of-the-mill computer and are consequently limited in their capabilities.

It is, therefore, an essential point of discussion, that how IoT devices can overcome their resource restrictions without giving up their private information [13]. Apart from all the aforementioned obstacles, there is also the inert concern of relying on centralized services. Dealing with the two aforesaid problems individually presents a divergent developmental trajectory, for instance if the focus is wholly placed on privacy, then realizing effective resource sharing might become untenable. On the other hand, focusing wholly on resource sharing might give rise to a scheme that sacrifices privacy for the sake of system throughput. It is therefore paramount to formulate a scheme that achieves a fine balance between privacy and effective inter-device communication.

Blockchain technology provides a trustless and decentralized environment for IoT Resource Sharing, but cannot guarantee privacy since public visibility is an essential security feature. Therefore, another groundbreaking technology, namely Zero Knowledge Proofs, can be used in tandem with blockchain to allow devices to operate with maximum anonymity while sharing resources with each other and keeping the intimate details of their sharing arrangements (such as compensation amounts) private from third parties. This very concept forms the working principle, and also constitutes the main novelty, of this research. The main contributions of this work are as follows :

demonstrate an IoT Resource sharing mechanism that utilizes Blockchain as a decentralized marketplace and observation tool.We exemplify transaction

privacy by shielding cryptocurrency amounts from third parties, using Zero Knowledge Proofs. Highlight the possibility of delegating blockchain data storage for lightweight devices by utilizing a fully synced, robust ethereum node, while also making use of a IPFS node for inter-device file-sharing. A unique protocol that focuses on privacy-preserved operation throughout, with device communication being provided through blockchain. With either device having no knowledge of the other device's intricate and personal information, this ensures anonymity. provide a holistic framework, complemented with cryptographic best-practices, which underscores the possibility for a secure and private resource / value exchange on a publicly visible, permissionless distributed ledger.

1.2 Background

1.2.1 Blockchain

A Blockchain can be described as a decentralized, distributed ledger technology that allows for secure and transparent record-keeping and verification of transactions. It consists of a network of computers, each of which maintains a copy of a shared digital ledger. When a new transaction is added to the ledger, it is verified by multiple computers on the network and then added to the blockchain, creating a permanent and unchangeable record of the transaction. This makes it difficult for transactions to be altered or tampered with, ensuring their security and integrity.

Blockchains themselves can be classified into two main categories: public or permissionless and private or permissioned. However, there are further variations, such as consortium and hybrid blockchains, but they are out of scope for the work being discussed. The choice of blockchain type depends on the specific requirements of a system, but all types operate on a peer-to-peer network and consist of nodes that maintain and update a shared ledger. Nodes possess the capability to verify transactions and initiate or receive processes independently.

Public blockchains are open to anyone and perform in a stable manner as long as majority of the clients follow security conventions, while private blockchains operate within a closed network. Since resource sharing is more suited to a public and auditable setting, we elected to go with a permissionless ledger (Ethereum) for this research work.

1.2.1.1 Ethereum

Ethereum [14] is a decentralized, open-source blockchain platform that enables the creation and deployment of smart contracts and decentralized applications (dApps). It was created in 2015 by Vitalik Buterin, a programmer and cryptocurrency researcher, and has since become one of the most popular and widely used blockchain platforms in the world.

One of the key features of Ethereum is its support for smart contracts, which are self-executing contracts with the terms of the agreement between buyer and seller being directly written into lines of code. This allows for the automation of complex and time-consuming processes, such as the execution of financial transactions or the management of supply chain logistics. Another key feature of Ethereum is its support for dApps, which are decentralized applications that are built on top of the Ethereum blockchain. dApps are typically open-source and allow for the creation and deployment of decentralized applications in a variety of fields, including finance, healthcare, and gaming. Ethereum is also notable for its use of its own cryptocurrency, called Ether, which is used to pay for transaction fees and services on the Ethereum network. Ether has become a popular digital currency, and is often used as a store of value or for trading on cryptocurrency exchanges.

Ethereum is often referred to as a "world computer" because it allows for the creation of decentralized applications that can be run on a global network of nodes, rather than being run on a single server or group of servers controlled by a single entity. This makes it more resilient to censorship and downtime. Overall, Ethereum is a powerful platform for building decentralized applications

and bringing new and innovative use cases to the world of blockchain technology.

Ethereum is the blockchain of choice for our work and the adjoining the Zero Knowledge solution, Tornado Cash, is also natively hosted on Ethereum. This makes the proposed work in this thesis a purely blockchain-driven, and fundamentally decentralized, privacy-oriented framework.

1.2.1.2 Smart Contract

A smart contract is a self-executing contract with the terms of the agreement between buyer and seller being directly written into lines of code. The code and the agreements contained therein exist across a distributed, decentralized blockchain network. Smart contracts allow for the automation of the contract execution, and they enforce the performance of the contract. This can help to reduce the need for intermediaries and can increase the security of the contract by making it tamper-evident.

1.2.2 Mixers

In the context of cryptocurrency, a mixer is a service that helps to increase the privacy of a transaction by mixing it with other transactions. This makes it more difficult for anyone to trace the transaction back to the original sender or recipient. Mixers are sometimes referred to as "tumblers" or "laundry services" because they can help to "clean" the history of a transaction and make it more difficult to track.

1.2.2.1 Custodial Mixers

A custodial mixer [15], also known as a centralized mixer or a mixing service, is a type of service that is used to mix different cryptocurrencies in order to obscure their original source or destination. This is often done in order to improve the privacy and security of cryptocurrency transactions, as it makes it more difficult for third parties to track the flow of funds.

Custodial mixers work by taking a large number of incoming transactions and mixing them together before sending the mixed funds to the specified destination

addresses. This makes it difficult for anyone to determine the original source of the funds or the ultimate destination of the mixed funds.

The main advantage of custodial mixers is that they can provide a simple and convenient way to improve the privacy of cryptocurrency transactions. However, there are also some disadvantages to using custodial mixers. For example, because the mixer holds the user's funds during the mixing process, there is a risk that the mixer could be hacked or go offline, resulting in the loss of funds. Additionally, some custodial mixers may charge fees for their services, which can reduce the overall profitability of the mixed transactions. such systems are often riddled with legal quandaries [1].

1.2.2.2 Non-Custodial Mixers

A non-custodial mixer, also known as a decentralized mixer or a trustless mixer, is a type of service that is used to mix different cryptocurrencies in order to obscure their original source or destination. Unlike custodial mixers, non-custodial mixers do not hold the user's funds during the mixing process, which reduces the risk of loss due to hacking or other issues.

Non-custodial mixers typically use cryptographic techniques, such as zero-knowledge proofs, to enable users to mix their funds without revealing their identities or the details of their transactions to the mixer. This provides a high level of privacy and security, as it ensures that only the user has control over their funds and information.

The main advantage of non-custodial mixers is that they provide a high level of privacy and security for cryptocurrency transactions. However, these strong privacy & security guarantees bring along some trade-offs with them. For instance, non-custodial mixers can be more sophisticated to use than custodial mixers, and they may require the user to have a certain level of technical expertise. Additionally, some non-custodial mixers may charge fees for their services, which can

[1] https://home.treasury.gov/news/press-releases/jy0768

might have a slight impact on the profitability of the mixed transactions.

Tornado Cash is one such non-custodial mixer and it was included in this research due to its strong privacy and anonymity guarantees. Since we integrate Tornado Cash into the code of the simulated IoT device, there is very little need for an owner of the IoT device to have technical know-how on Tornado and most of the procedure is automated by the IoT device itself.

1.2.2.3 Tornado Cash

Tornado Cash is a decentralized finance (DeFi) platform built on the Ethereum blockchain. It is a non-custodial and anonymous Ethereum mixer, which means that it allows users to mix their Ether (ETH) and ERC-20 tokens in a way that hides their transaction history and makes it difficult to trace the funds back to their original owner.

Tornado Cash uses a technique called zero-knowledge proofs to enable anonymous transactions. This allows users to send and receive funds without revealing their identity or the details of the transaction, making it a privacy-focused DeFi platform. In addition to providing anonymous transactions, Tornado Cash also allows users to earn interest on their funds by providing liquidity to the platform. This makes it a popular choice for users who want to earn passive income on their cryptocurrency holdings while also maintaining their privacy.

Tornado Cash is the platform of choice that is used by the presented work to protect inter-device payments through zero-knowledge and ensure that payment amounts stay hidden from prying eyes. It has also seen usage in prior academic work [16, 17] and is justifiably a suitable option for being one of the core modules for this work.

1.2.3 Zero Knowledge

Zero knowledge proof (ZKP) is a method by which one party (the prover) can prove to another party (the verifier) that they know a certain piece of information, without revealing the actual information itself. This is achieved through the use

of a special type of cryptographic protocol, in which the prover and the verifier engage in a series of interactions that allow the prover to convince the verifier that they possess the information in question, without revealing any details about the information itself. This concealment of the information, which is being proved, is the bedrock of the strong privacy prospect offered by ZKPs and as such forms a core part of the work being presented in this thesis.

Shafi Goldwasser et al. [8] conducted the first such research which explored the idea of interactive proof systems, which are a type of computational procedure in which a prover tries to convince a verifier that a given statement is true. This thesis discusses the idea of knowledge complexity, which is a measure of how much knowledge the prover must have in order to convince the verifier of the truth of the statement. This work shows that the knowledge complexity of interactive proof systems can be much smaller than the computational complexity of the problem being solved, which means that interactive proof systems can be much more efficient than other computational methods. This paper is considered as the precursor of all subsequent zero-knowledge cryptography and zero-knowledge systems that followed suit afterwards. SNARKs, detailed in the following section, was a direct consequence of this paper.

With the passage of time and advances in computational power as well as in cryptographic sciences [18], new and improved techniques came into play. Among them were STARKs, SNARKs and the very application of one of these proofs, that is being utilized by the presented work i.e Tornado Cash.

1.2.3.1 SNARKs

SNARKs, or Succinct Non-Interactive Arguments of Knowledge, are a type of cryptographic proof that allows one party to prove to another party that they know a certain piece of information, without revealing the actual information itself. SNARKs are zero-knowledge proofs, meaning that they provide a way for one party to prove that they know a certain fact without revealing any additional information beyond the fact itself. SNARKs are useful in many applications, including in

blockchain technology, where they can be used to enable private transactions on a public ledger.

1.2.3.2 STARKs

STARK (Scalable Transparent ARgument of Knowledge) is a type of zero-knowledge proof, which is a way to prove the truth of a statement without revealing any additional information beyond the fact that the statement is true. zk-STARKs are particularly notable for their short proof sizes, high speed, and high security, making them a promising option for use in a variety of applications.

STARKs were first mentioned in the research done by Ben-Sasson et al. [19]. It proposes a new approach to maintaining the integrity, security, and functionality of computational systems in a way that is scalable, transparent, and secure against potential attacks from quantum computers. Ben-Sasson's approach is based on the use of a novel type of cryptographic protocol called "proof-of-replication." This protocol allows a computational system to verify the integrity and correctness of its operations and provides a high level of security against potential quantum attacks. STARKs, at the time of writing, constitute a very new technology with an ever-changing and volatile technical ecosystem.

1.3 Thesis Outline

The rest of this document is presented as follows. In Section 2, we present a detailed review of the relevant literature. Section 3 describes the detailed methodology of the entire protocol along with in-depth description of each and every component. Section 4 discusses the acquired results, the evaluation metrics along with the comparative performance of a competing approach. Finally, Section 5 concludes this thesis with the outcomes of this research work accompanied by suggestions for prospective future work.

CHAPTER II

LITERATURE REVIEW

Being an emerging field, there is a sizeable amount of work already present for IoT, resource sharing and blockchain. We present a simple comparison of our research with a handful of notable prior works, which are presented in a demarcated style of three subcategories. These are dedicated to works focusing just on IoT Resource sharing, IoT resource sharing with Peer to Peer mechanisms & lastly IoT resource sharing with blockchain implementations.

We start off with general resource sharing protocol that is proposed through the IoT paradigm. In [20], the research proposes a resource sharing approach over a centralized management architecture. A new protocol announcement message is suggested, called the DFSP or Function and Service Discovery Protocol, which operates over the MQTT protocol. An AI controller operates at the backend & uses Machine Learning techniques to allocate resources. The researchers concluded that by using the length of the payload of PUBLISH message, it is sufficient to effectively convey the information about the functions and services offered by a Thing in an IoT network. In [21], a comprehensive trust management system is proposed by the name of ShareTrust. It is tailor-made to provide trust computation and aggregation for resource sharing among IoT devices in an IIoT setting. The reputation & trust rating of providers as well as those of the seekers is heavily subject to their behavior in the network and a rigorous mathematical model is used to evaluate the trustworthiness and stability of both. Despite boasting an extensive framework, the ShareTrust solution is openly advertised as being heavily centralized. Consequently, it lacks the high availability and distributed robustness of a peer-to-peer system like a blockchain. It also does not provide any solution for the privacy preserving aspect between the IoT devices. Lastly, In [22] a method

called SP-DPM is proposed. The core purpose of the work is to preserve security and privacy of IoT devices in the cloud environment. The proposed method uses techniques such as k-anonymization, clustering, and encrypted analysis to partition, partially decrypt, and analyzes data in a way that improves the model's efficiency while maintaining security. The authors conduct experiments and claim that the proposed method achieves high accuracy, precision, recall, and F1-score while improving on the results of state-of-the-art methods. The work does not use any kind of decentralized model of operation (like peer-to-peer or blockchain) and relies on a cloud infrastructure, which makes it a naturally centralized proposition.

With advances in technology, peer-to-peer protocols became increasingly invited more research & innovation in concert with other emerging technologies, with IoT being one of them. SeCaS, proposed in [23], introduces an extensive peer-to-peer procedure for capability sharing among IoT devices. Additionally, the research also details the protocols that provide identification of the devices' network as well as an authentication mechanism for messages that being exchanged among devices. The technical ecosystem, within which the devices operate, thus effectively constitutes a decentralized environment managed through Distributed Hash Tables (DHT) [24, 25]. Subsequently, in [26] the research aspires to develop a system based on the notion of decentralized private-by-design IoT. The core principle is that data produced by personal IoT devices will be safely archived in a distributed system which guarantees privacy through it's design & allowing the owners of the data to be the main stakeholders for the decision of whether to share their data and with whom. To achieve this goal, a the use of Peer-to-Peer storage networks is suggested. They propose the use of a least authority file system, namely Tahoe-LAFS [27] as a peer-to-peer file storage. However, since their method lacks an active entity, like a blockchain, gate-keeping the data append operation, the system is potentially open to attack. Attackers can easily clog the system with obsolete data. In [28], a socially-aware scheme is used for efficient video stream through a peer-to-peer approach. The research suggest using social

links, like friendships on social media, to optimize video transmission. The policy employs weighted fair queue (WFQ) for video transmission. Queuing priority is based on classes of different relationships, since each peer will have a unique friend list and the social links are ordered by priority based on the type of relationship like family, friends and so on. The research aspires to promote mutual resource sharing and provide users good video sharing and watching experiences with multimedia IoT devices. However, absence of a conscious blockchain network can open up the system to a plethora of dangers, including but not limited to, privacy violations & people pretending to be friends but are malicious impostors on hacked social media.

After the advent of bitcoin [29], blockchain & distributed ledgers (DLT) became more influential within the sphere of technology. Naturally, it incited researchers to combine disruptive technologies and resultantly a great deal of recent literature details the use of IoT with blockchain technology. In[30], the author suggests a large scale IoT network with numerous provisions and procedures in place to ensure quick and seamless connectivity for incoming devices which is labelled as network on demand(NoD) as well as an intricate resource sharing mechanism. Devices can join and be registered into the network through a comprehensive procedure. Monitor nodes are used to oversee a resource sharing deal and update the reputation of providers accordingly. In case of a mishap from the provider, reputation is severely deducted or even wiped off, following a very stringent security scheme. Although the publication defines an extensive protocol, it provides no provision for size of blockchain data, instead raising the increasing size as an Open Research Challenge. We handle this using a combined solution comprising of decentralized storage and a fully synced Ethereum node, that is accessible by the IoT device itself. Additionally, it does not mention any type of privacy preserving provisions to protect transacted amounts from prying eyes. We use an on-chain solution, embedded within the ethereum ecosystem, to especially provide a fast and scalable zero knowledge facility to keep transactions truthful yet private.

Similarly, in [31] a combination of Artificial Intelligence with Distributed Ledger is proposed in order to produce a resource sharing platform which is geared towards the application of Industrial IoT (IIoT) scale. An adaptive learning approach is utilized to design a trust management system for recording reputations of different IIoT devices which is managed through network edges and maintained in distributed ledgers. A social-aware device selection model is proposed, which is utilized for selecting a trustworthy IIoT device to offload tasks to and hence share it's resources for a particular task. The work is based on PoET (Proof of Elapsed Time) consensus, which requires specialized and trusted hardware to run. Our thesis has no such bindings and is comparatively flexible in this regard.

A tethered network of between blockchain and IoT is demonstrated by work presented in [32], where a local peer (lpeer) on the network communicates with a peer present on-chain (anchor peer) to manage IoT requests to the blockchain. A Local Peer Network is used to segregate IoT requests from the main blockchain network in order to avoid from choking with high traffic. The IoT device communicates with a blockchain sdk, which is interfaced to the local peer network and this allows the lpeer to gather all Iot requests, order them and then send them to the anchor peer on the blockchain. All the keys, authentications and digital signatures involved are issued by a certificate authority (CA), which itself is housed inside the local peer network. Since this work involves the usage of a CA, it adds a layer of centralization to their implementation. Our proposition has no such requirements. Moreover, their work hinges on the communication and longevity of two peers which reside on blockchain and off-chain and work to connect a localized network to the blockchain. Failure of either would cut off their system from blockchain. Our system has no such dependencies.

The work presented in [33] suggests an approach towards a resource sharing scheme by using worm computing intertwined with blockchain. They adopt a two-fold approach, where nodes are either involved in resource sharing or securing the network. Participants can either be users or sharers of resources and can pay for using

resources or get paid for offering resources. Additionally, nodes can choose to serve the purpose of securing the network by sharing network defense information. Both the operations are recorded on the blockchain and the transactions are mined to maintain a track record of all activity. The approach lacks transaction privacy and despite implementing an exhaustive security defense module to protect the network, there is no mention of privacy-preservation throughout the work.

CHAPTER III

PROPOSED WORK

3.1 Proposed Working : ZeKSA

We condense our work by giving it a abbreviated moniker, namely **ZeKSA** (Zero-knowledge Secured Anonymous IoT Resource Sharing on Blockchain). A high level and summarised overview of ZeKSA's system architecture is shown in Figure 1. In the subsequent portions, we discuss the different actors in the system and their purpose in making ZeKSA work.

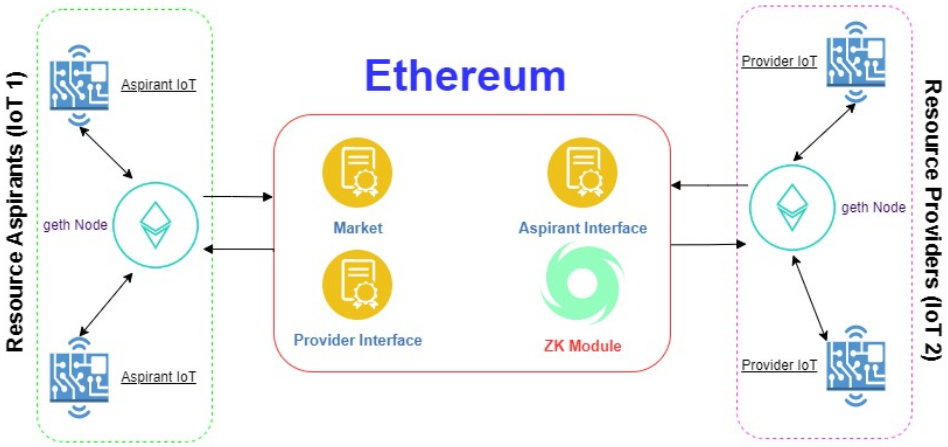

Figure 1: Overview of System Architecture

3.2 Components

ZeKSA is based on several components, namely Network Participants & Smart contracts which serve as Interfaces for any device that partakes in the ZeKSA protocol

3.2.1 Network Participants

A brief description of the network participants is given as follows;

1. **Provider IoT :** This device type is a service provider and is supposed to have one or more useful resources for others to use. This device listens to a specific smart contract interface, as it aims to detect any suitable resource requests on the network. It holds a blockchain wallet to accept payments and it's business logic on the platform is tethered to a ethereum smart contract known as the Provider Interface, which is detailed in the following subsection.

2. **Aspirant IoT :** This device acts as a consumer and will mostly be requesting resources from provider devices. It holds a blockchain wallet to issue zk-secure payments to respective service providers. This device will send out requests into the network through messages encoded in transactions and will expect offers to arrive from prospective Providers. Its on-chain logic is laid out in a smart contract known as the Aspirant Interface, which is detailed in the following subsection.

3.2.2 Interfaces

The following section concisely details the smart contracts, which serve the purpose of interfacing the devices with the blockchain, involved in this protocol;

1. **Market :** This is a smart contract that serves as a matchmaking interface for aspirant and provider devices. It contains the logic for broadcasting the resource requests of aspirants on the blockchain. Additionally, providers can listen to this market interface and consequently pick out requests that they might be potentially capable of fulfilling.

2. **Provider Interface :** This is a smart contract that contains subroutines for the acceptance of a proposed resource provision offer by a potential customer(aspirant) device, for exchanging workloads and their computed outputs between aspirants & providers and for receiving zero-knowledge payments from aspirant devices upon successful completion of a resource sharing task. Both, the aspirant and provider devices actively interact with this interface.

3. **Aspirant Interface :** This is a smart contract that contains the logic for an aspirant device to invite offers from possible provider devices, for consideration and subsequent approval or rejection. Provider devices frequently interact with this interface, while the respective aspirant devices usually use this interface to listen and filter among offers from providers.

3.2.3 Additional Modules

Additional components, which are neither off-chain IoT devices nor can they be classifed as defacto smart contracts, are briefly discussed below;

1. **ZK Module :** This is a decentralized application that operate on the blockchain and secures the privacy of payments between two devices by obfuscating the transaction amounts. It does so by taking in a specified amount of cryptocurrency from the invoking device and issuing a zero knowledge secured 'note' in exchange. This note can later be redeemed for the amount which was originally deposited in order to obtain this note.

3.2.4 Device Entry:

For joining a network, a new device undergoes an extensive yet swift procedure as detailed in Figure 2.

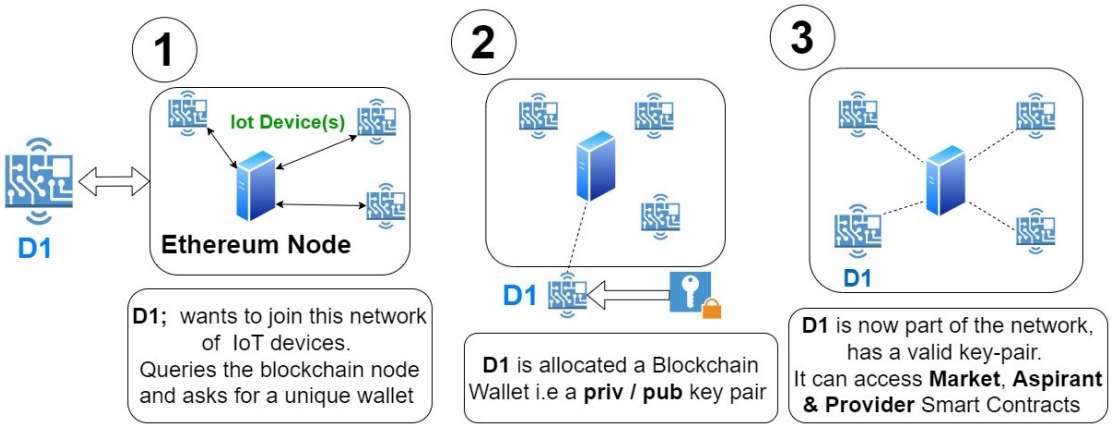

Figure 2: IoT Device Entry into the Network

Regardless of its role, whether aspirant or provider, we assume an IoT device **D1** wanting to join a network of devices. The very first action the device takes

is to announce it's intentions for joining the network. This is done by the device querying the closest Go-Ethereum (geth) [34] node for a wallet creation. After this, a cryptocurrency wallet is created and allocated to the device. This means that the device is now in possession of a private/public key pair and has a valid, verifiable presence on blockchain.

3.2.5 Device Perspectives

The proposed protocol, at its core, revolves around the principle of active interaction between two devices, while preserving the privacy of either participant and additionally keeping the payment amounts between the two actors private from any observing third party. This ensures that while sensitive information is kept private, there is still enough evidence visible to the public eye that any discrepancy, from either device, can be easily identified. In order to properly understand ZeKSA's operation, we need to view the system from the vantage points of the two main actors in the system, which happen to be the Aspirant and Provider devices respectively.

3.2.5.1 Aspirant view

The aspirant's reference point and interactions with the system are given in Figure 3.

The aspirant device, therefore, strives to use resources on offer from other devices in exchange for cryptocurrency payments. It achieves this by broadcasting its needs through a blockchain transaction on the market interface, by calling a smart contract method that encapsulates its request and broadcasts it throughout the blockchain network. This request, is propagated in the form of an Ethereum Event [1], and is synonymous with the corresponding transaction. Since events are used for logging purposes in ethereum, the principle holds that an event can only be fired if a transaction is carried out to do so. This concept of communication with the help of events is widely and extensively used throughout the protocol.

[1] https://solidity-by-example.org/events/

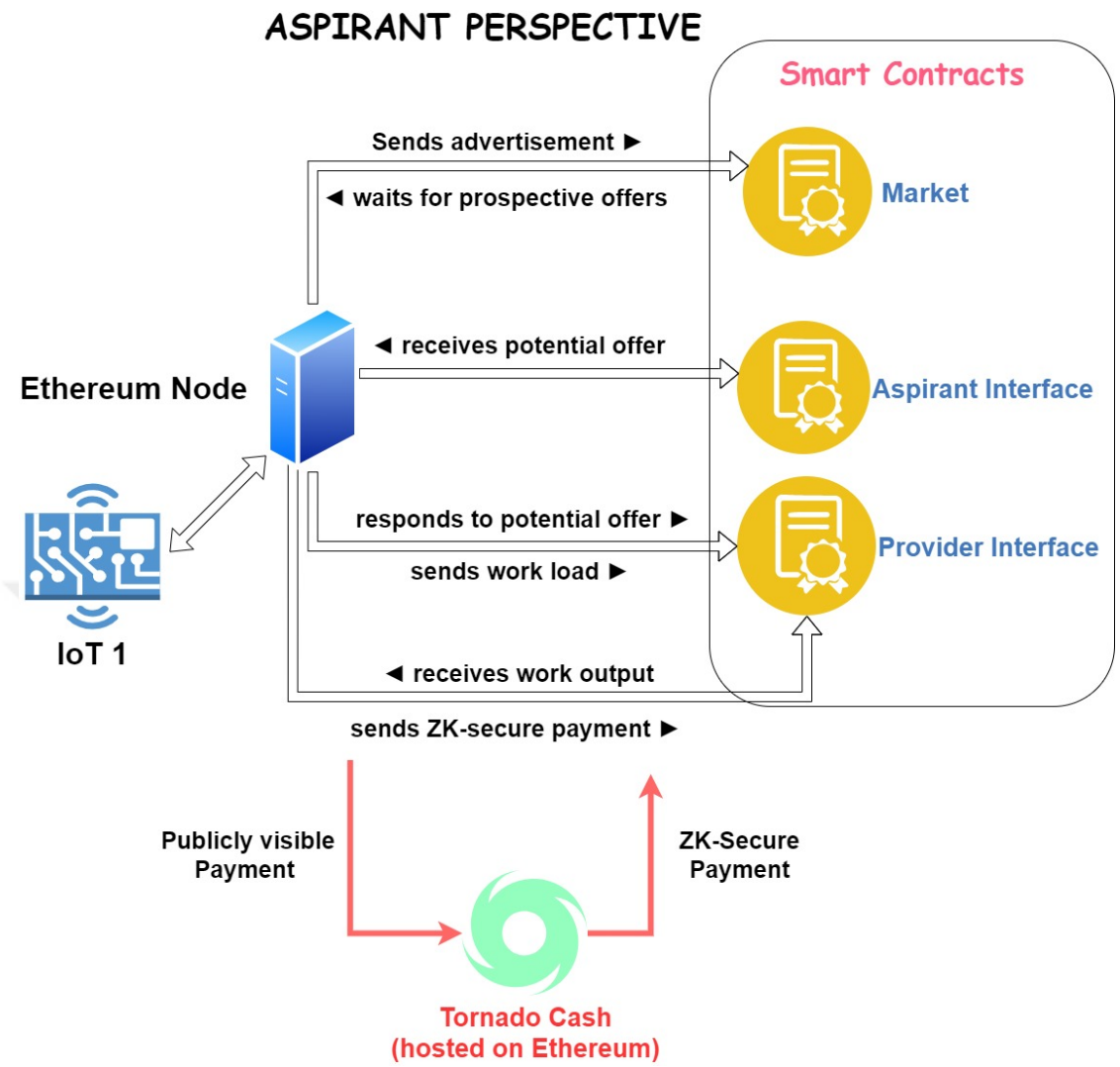

Figure 3: Aspirant

Apart from broadcasting it's demands, the aspirant also listens to its own (Aspirant) interface for any incoming offers. Offers received by an aspirant device are also in the form of events, where the address of the provider interface is also communicated for communication purposes. Upon receiving an offer, the aspirant device can either accept or reject the offer by calling the appropriate method in the interface of the corresponding provider device. If it decides to accept an offer, then the aspirant calls the respective offer accepted method in the provider interface. It then sends the workload, for whose execution/processing it needs a particular resource, through the provider interface. After the provider has performed work and executed the workload, a result is returned to the aspirant through the

provider interface. The aspirant can then invoke the zk-module and submit the agreed upon payment amount to the module, to secure it using zero-knowledge. A zk-secure redeemable payment note is returned to the aspirant device. This note is then encrypted using the providers public key and sent to the provider, again through the provider interface.

3.2.5.2 Provider view

The other main participant is the provider device, whose vantage point in the protocol is exhibited by Figure 4.

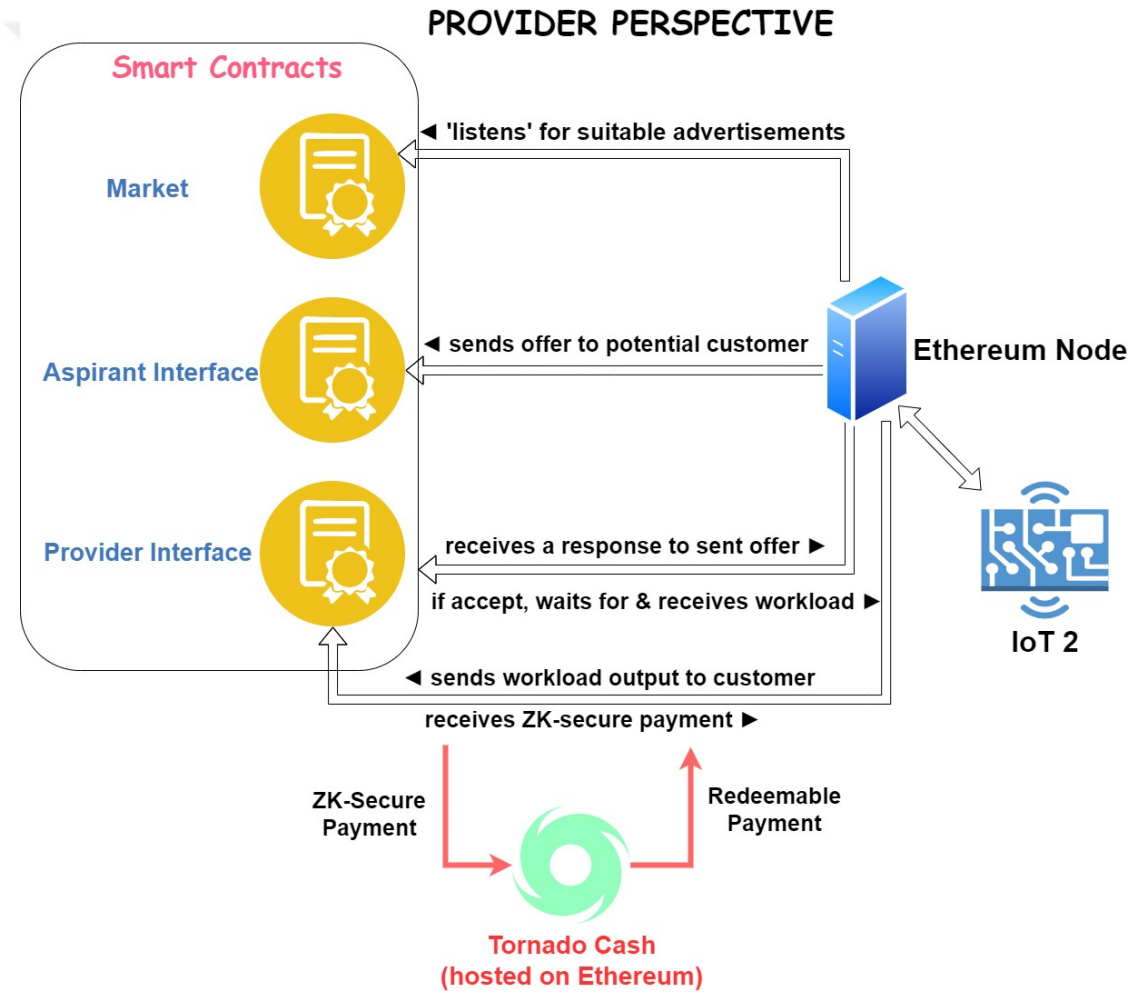

Figure 4: Provider

As evident by its moniker, this device aims to provide its resources to aspirant devices on the network. In exchange for its services, it receives a cryptocurrency

payment. A provider device, when idle, listen to the market interface for any suitable resource request that it may be able to fulfill. If it detects a suitable request, which will be present in the events that are fired by aspirant devices through the market interface, it quickly formulates an offer and then sends an offer to the corresponding aspirant. It does so with the help of the information that is included in the event, which allows it to locate the address of the specific aspirant's interface and call the appropriate method for relaying the offer. It then waits for a response from the aspirant. The provider will wait until it receives a response from the incumbent aspirant or until the timeout period is reached, after which it starts listening to the market interface again for another suitable resource request. In the situation that the offer is accepted, the provider has to wait for a workload, which is delivered to the provider interface by the aspirant. The provider device can then retrieve the workload, execute it using its own resources and obtain the workload output. After carrying out the work, the output result is then delivered to the aspirant, through the provider interface. The provider then waits for the payment, which is delivered to it through the provider interface. Once the payment is received, the provider has to decrypt the payment using it's own private key. In its decrypted form, the payment can be submitted to the zk-module and be redeemed for the agreed upon amount of cryptocurrency.

3.2.6 Zero Knowledge Module

The zero-knowledge (zk) module is arguably one of the most crucial part of this work, as it focuses on privacy preservation of this protocol. Although cryptocurrency obfuscation techniques are often confused with purely custodial mixing services [15], such systems are often riddled with legal quandaries [2] and can even hold certain security/privacy risks owing to advanced de-anonymization techniques as elaborated in [35]. We utilize a slightly different approach, courtesy of the Tornado Cash (TC) [36] solution. While the cryptographic details are explored in depth in

[2]https://home.treasury.gov/news/press-releases/jy0768

[37], elucidating them is out of the scope of this work. Instead, we summarise the core working elements of TC, which make it a feasible inclusion in our protocol and contributes substantially to the security offering of the proposed work.

Most of the available mixers are custodial [3], which means that they take full ownership of the users' cryptocurrency when obfuscating a transaction. This introduces an element of dependence, where the user has to trust whoever is running the platform, with their money. There is a risk that the user's funds might be misappropriated, stolen or lost altogether since the custody of the funds is transferred from the user to the service operator. In sharp contrast, a non-custodial mixing service allows the user to retain full control of their funds, even when using the service. TC happens to be a non-custodial mixer and consequently, it allows the IoT devices in our protocol to be in full control of their funds when they are utilizing the service to anonymize their payments.

Tornado Cash utilizes smart contracts which are called pools. These pools allow users to transact privately on Ethereum. When prompted, pools will carry out one of two supported operations, either a **deposit** or a **withdraw**. These operations allow a user to deposit tokens from one address and later on withdraw those same tokens to a different address. And even though all events occur publicly on Ethereum's public transparent ledger, any public link between the deposit and withdrawal addresses is broken, thus ensuring user privacy. Users can withdraw and use their funds without fear of their financial records being exposed to third parties. Tornado Cash pools enforce a strict rule, which stipulates that users can only withdraw the specific tokens they originally deposited. This ensures that Tornado Cash pools are entirely non-custodial. So, a user who deposits and later withdraws tokens maintains full custody and control of their tokens, even as they pass through the pool. At no point during the entire process is the user required to relinquish control of their tokens to anyone else. The reason the deposit and withdraw functionality of TC holds strong is because of zero-knowledge cryptography,

[3]https://blockworks.co/news/crypto-mixers-and-privacy-coins-can-they-resist-censorship

which is part of Tornado Cash's smart contract code, and forms the foundation on which the deposit and withdrawal operations function.

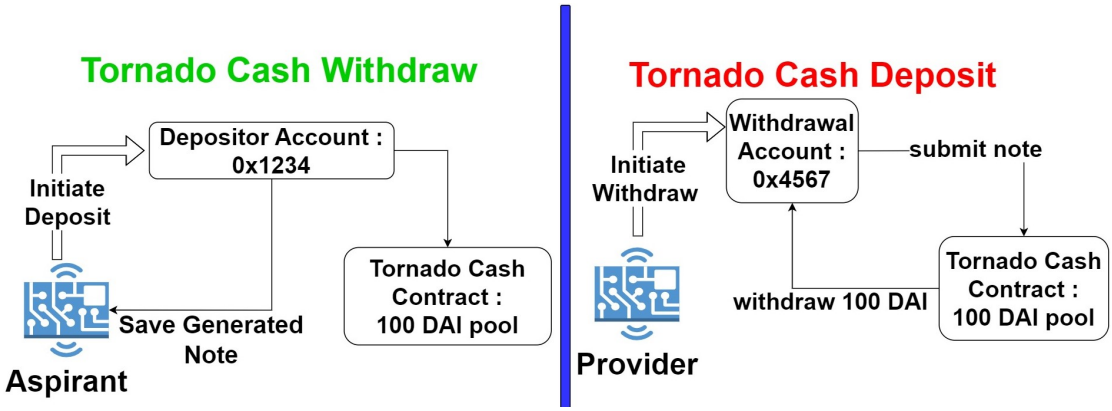

Figure 5: Depositing/Withdrawing with Tornado Cash

As shown in Figure 5, when 100 DAI [4] tokens are deposited on the TC contract, a commitment hash has to be provided. This commitment hash will be stored in a Merkle tree by the TC smart contract. This hash is generated through a locally note is locally generated, which ensures privacy and safety of the note. This note will be instrumental in withdrawing these same tokens later on.

In the same Figure 5, these tokens are withdrawn to a different account. For this to go through, 2 zero-knowledge proofs must be provided. The first proves that the Merkle tree contains the commitment that was provided when depositing these tokens. And because the user should only be able to withdraw their deposit only once, they also have to provide a nullifier that is unique for the commitment. The uniqueness of the nullifier is ensured by how the commitment is generated during the deposit. The commitment is generated from the nullifier and a secret by hashing. Both, the nullifier and the secret, are actually embedded in the note that was generated during the deposit process. Since hashing is a one-way function, it is impossible to use the commitment to figure out the nullifier or the secret. Since the entire security of funds is contingent on the note, it is evident that whoever holds the note also holds the access to the deposited funds. This makes TC a reasonably private medium of payment to be utilized by the proposed protocol.

[4] https://makerdao.com/en/

A recent analysis [38] of TC's privacy offering concluded that the only hindrance in guaranteeing absolute secrecy through the platform is through human error i.e naive and repeated activity like immediate withdrawal of funds after depositing into tornado cash pools, multiple & rapid deposits of the same currency denomination stand to give away slight clues about usage and user patterns. The platform, on its own, guarantees enough privacy and resilience to merit inclusion as an important component of our work.

3.3 Protocol

Keeping in mind the concepts of the different component, we now present a complete workflow of the protocol technical & operational apparatus with an example use case. The entire procedure is detailed in Figure 6.

The scenario assumes that an aspirant device is looking for a CPU-intensive task for generating a random number and a provider device is listening to the market interface. Therefore, according to the sequence diagram, the message **1** underlines that a provider device is actively listening for any incoming resource requests. In message **2**, the aspirant sends a service request to the market interface on the blockchain. This is picked up by the provider in message **3**. Meanwhile in message **4**, after sending the request the aspirant waits for prospective resource propositions to come in by listening to its own interface on the blockchain. In message **5**, the provider formulates an offer and sends it by calling the corresponding method in the aspirant's interface. In message **6**, the aspirant is made aware of the incoming offer by its interface and consequently proceeds to analyze the offer. In the meantime, in message **7**, the provider now starts listening to its own interface for a probable response from the aspirant device. The aspirant decides to accept the offer, and calls the appropriate method in the provider's interface as shown by message **8**. This alerts the provider, which now braces itself for incoming workload in message **9**. The aspirant then uploads the workload to IPFS and encrypts the

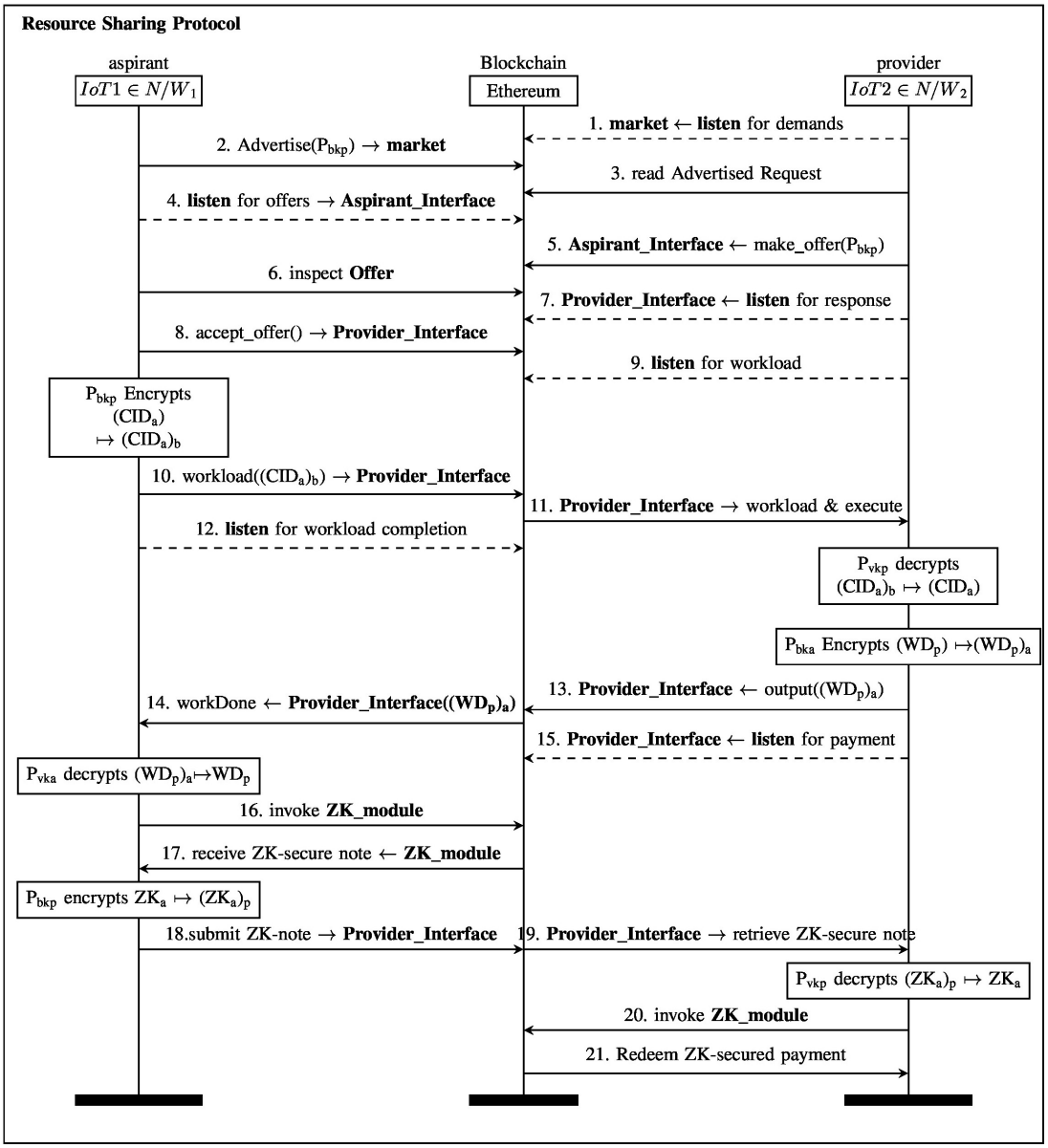

Figure 6: Resource Sharing protocol

obtained IPFS content address[5] workload using the public key of the provider and delivers it to the provider interface in message **10**. Eventually, the provider is able to acquire the encrypted content address through its interface. After decrypting the obtained content address with its own private key, the provider is able to acquire the workload from IPFS. It subsequently executes the workload as shown in message **11**. In the meantime, the aspirant listens to the provider interface, in message **12**. In due course, the provider delivers the results of the execution

[5]https://docs.ipfs.tech/concepts/content-addressing/#what-is-a-cid

through its own interface in message **13**. The execution output is obtained by the aspirant, after querying the provider interface in message **14**. The provider, meanwhile, listens to its own interface for payment notification **15**.

As detailed in the previous section, the aspirant now makes use of the zk_module to anonymize the payment amount in message **16**. The zk_module returns a zero-knowledge secure note to the aspirant in message **17**. This note is then encrypted by the aspirant with the provider's public key and sent to the provider interface in message **18**. In message **19**, the provider is able to receive the encrypted and zk-secure note. Then, in message **20** the provider decrypts the note, using its own private key. Since it now holds the unencrypted note, this indicates that it now owns the funds that were deposited in exchange for this note. Now, the provider invokes the zk_module. In message **21**, the zk_module, after verifying the proofs embedded in the note, transfers the corresponding amount of tokens to the wallet of the provider device.

For this workflow, we take inspiration from the work done by Angeliki et al. [23], which also demonstrates resource sharing in a decentralized environment powered by Distributed Hash Tables (DHT).

CHAPTER IV

PERFORMANCE EVALUATION

We evaluated our work by comparing it with the most similar work available throughout the latest, state-of-the-art literature. The closest solution that could be considered as a competing protocol, was IoTShare [30]. Before explaining our methodology of comparison, we explain the competing protocol in the following section.

4.1 IoTShare

IoTShare is a research which details a blockchain-based platform for enabling the secure and efficient sharing of IoT data and resources among multiple parties. The platform, called IoTshare, uses a decentralized architecture and cryptographic protocols to ensure the privacy, security, and integrity of the data and resources shared on the network. IoTshare allows users to securely share data and resources with other users, while maintaining control over the access and usage of their data and resources. The work also discusses the potential applications and benefits of using IoTshare in smart city situation-awareness applications, such as traffic management, public safety, and environmental monitoring. Overall, the paper presents IoTshare as a promising solution for enabling the collaborative development and use of IoT applications and services in smart cities.

Basically, IoTshare uses blockchain technology to provide a decentralized and secure platform for sharing IoT data and resources. The platform uses a distributed ledger to record and manage the transactions and interactions among the users on the network, ensuring that the data and resources shared on the network are tamper-resistant and transparent. IoTshare also employs cryptographic protocols to protect the privacy and security of the users and their data and resources. The platform uses digital signatures and encryption to verify the identity

and authenticity of the users, and to ensure that the data and resources shared on the network are only accessible to authorized users. IoTshare provides a range of tools and services for managing and monetizing IoT data and resources. For example, the platform allows users to store, analyze, and visualize their data, and to set rules and conditions for sharing and accessing their data and resources. The platform also provides a marketplace for users to buy and sell their data and resources, enabling them to generate revenue from their data and resources. It also entails the discussion about potential applications and benefits of using IoT-Share in smart city situation-awareness applications. By enabling the secure and efficient sharing of IoT data and resources among multiple parties, IoTshare can help to improve the accuracy and timeliness of the data used in these applications, and to reduce the costs and complexity of managing and using the data.

4.2 Comparative Assumptions

While it is unfeasible to expect to completely reproduce another work without the input and the resources of the original authors, the IoTShare protocol was carefully studied, reviewed and subsequently analyzed in order to prepare a strong adn comprehensive evaluation baseline for ZeKSA. Below, we list some assumptions which we believe are vital to mention with regards to evaluations;

1. ZeKSA is a micro-level protocol, focused on the intricate details of secure interaction between two devices. IoTShare, in contrast, can be considered as a macro-level protocol

2. IoTShare explains many aspects of inter-device interaction, but does not mention user privacy anywhere throughout the work. ZeKSA, on the other hand focuses primarily on keeping device identities as anonymous as feasibly possible

3. Zero Knowledge is one of the bedrocks of ZeKSA's core system of operation, while IoTShare researchers did not use it anywhere throughout their research

4. ZeKSA worked with a fully synced local geth node, which was readily available to for making rpc calls to the ethereum testnet (Göerli network). IoTShare did not list the usage of such a facility anywhere, so we assume it did not use a fully synced node.

Therefore, IoTShare was reproduced with all the aforementioned assumptions intact and all the basic protocol functionality present. As per the assumptions, IoTShare did not leverage zero-knowledge secured payments and did not have a fully synced node to make procedure calls to the blockchain. Instead it used a thirdparty rpc [1] and made payments through direct ethereum transactions through a publicly exposed smart contract escrow system.

With the assumptions out of the way, we now proceed to describe the evaluation results.

4.3 Evaluation & Results

4.3.1 Setup

Our setup includes a fully synced geth node on Ubuntu 22.04 (Jammy Jellyfish) [2], working with a fully functional consensus client known as lighthouse [39]. We use the Göerli [3] testnet on ethereum, to evaluate our work in an environment that is similar to the ethereum main-net.

We compared our work to a variant of the IoTshare [30] work, previously discussed in section 2. To ensure a fair and concrete comparative evaluation, the IoTshare variant was studied in-depth. The architectural features of IoTshare, which directly represent counterparts of ZeKSA, were meticulously programmed in the exact same smart contracts as our own work. The IoTShare variant was also assigned aspirant & provider devices. The only differences, identified after studying the IoTshare work in detail, were that IoTShare did not use a fully synced node and that it made no attempt to implement privacy-preserving features for

[1] https://www.infura.io/
[2] https://releases.ubuntu.com/22.04/
[3] https://goerli.net/

inter-device payments. Therefore, it was assigned a third party remote procedure call (rpc) instead of a local node and its payment transactions were not allocated the facility of zero-knowledge protection, leaving it's transaction amounts open to public view and scrutiny.

4.3.2 Metrics Used

We elected to go with the following metrics to compare ZeKSA with IoTShare;

1. Execution Time : time taken for an entire workflow of resource sharing (with and without ZK) to execute from start to finish.

2. Gas usage : "Gas" is a unit of computation [4] which signifies the amount of computational power required to execute specific operations on Ethereum. Therefore Gas usage is a sensible metric to gauge the computational power consumed by our proposed workflows.

3. Transaction cost : each interaction with the Ethereum blockchain, that alters the state of the blockchain, demands that the invoking entity send a transaction. Even those transactions that don't involve any cryptocurrency transfer between the transacting entities, still consume some ether since such transfers require Gas to carry out computational instructions on the blockchain. Each transaction, therefore, commands a gas price which is used to carry out the instructions and consequently incurs a cost to the calling entity. Therefore, the cost of a transaction provides an insightful glimpse into the complexity of the work done through the transaction itself and serves as a significant metric for testing out ZeKSA.

4.3.3 Evaluations

4.3.3.1 Execution Time

We repeat the protocol workflow, as discussed in section 3.3, several times. We do the same with the IoTShare variant. We compare the execution times between

[4] https://ethereum.org/en/developers/docs/gas/

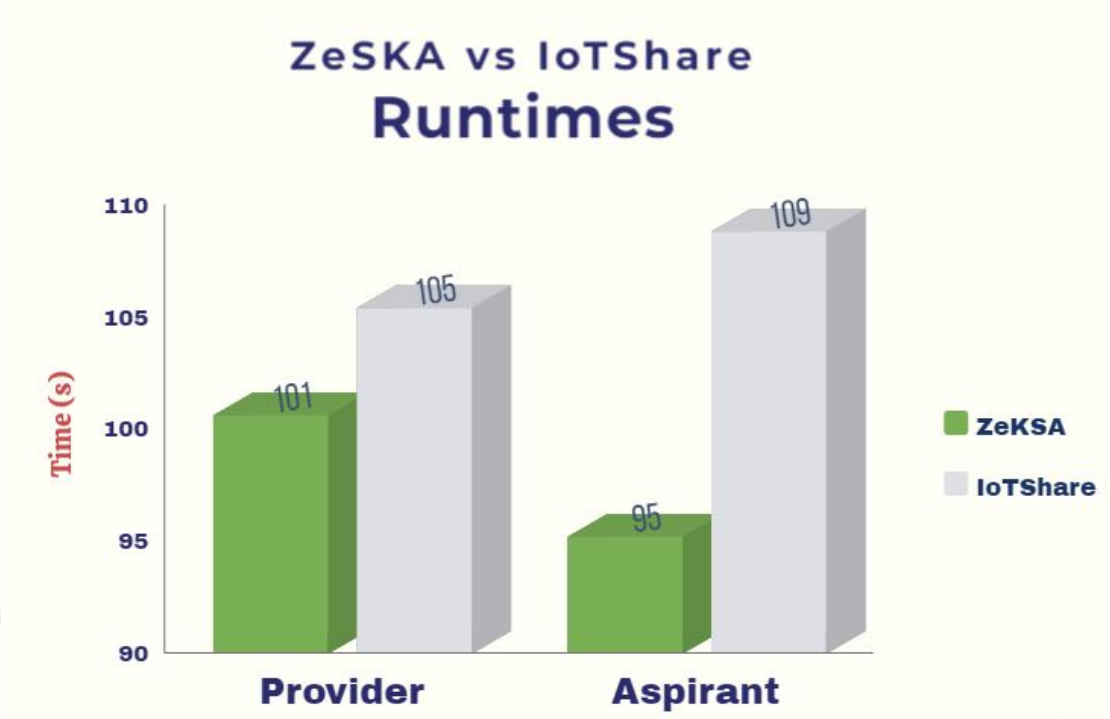

Figure 7: Execution Time Comparison

the respective aspirant and provider devices.

As shown in figure 7, ZeKSA is comparatively faster in terms of Aspirant execution times. It attains comparatively faster runtimes as compared to the competing approach which does not use zero-knowledge protection. The IoTShare approach, in comparison, takes more time to run completely. In terms of execution time, the ZekSA aspirant device is 12.51% faster than the non-secure approach, on average.

For provider devices, it is evident from figure 7 once again that, ZeKSA is once quicker on average. Provider devices of ZeKSA exhibit a faster execution time on average, holding a slender advantage of around 4.55% over its IoTShare counterpart.

4.3.3.2 Gas Usage

To gauge the computational strain on the blockchain for both approaches, the one with zero-knowledge and the one without, we record Gas usage for both, aspirant

& provider devices, with the help of Etherscan [5].

First off, we observe the Gas usage patterns for the provider devices, since they perform most of their grunt work off-chain (i.e workload processing / execution).

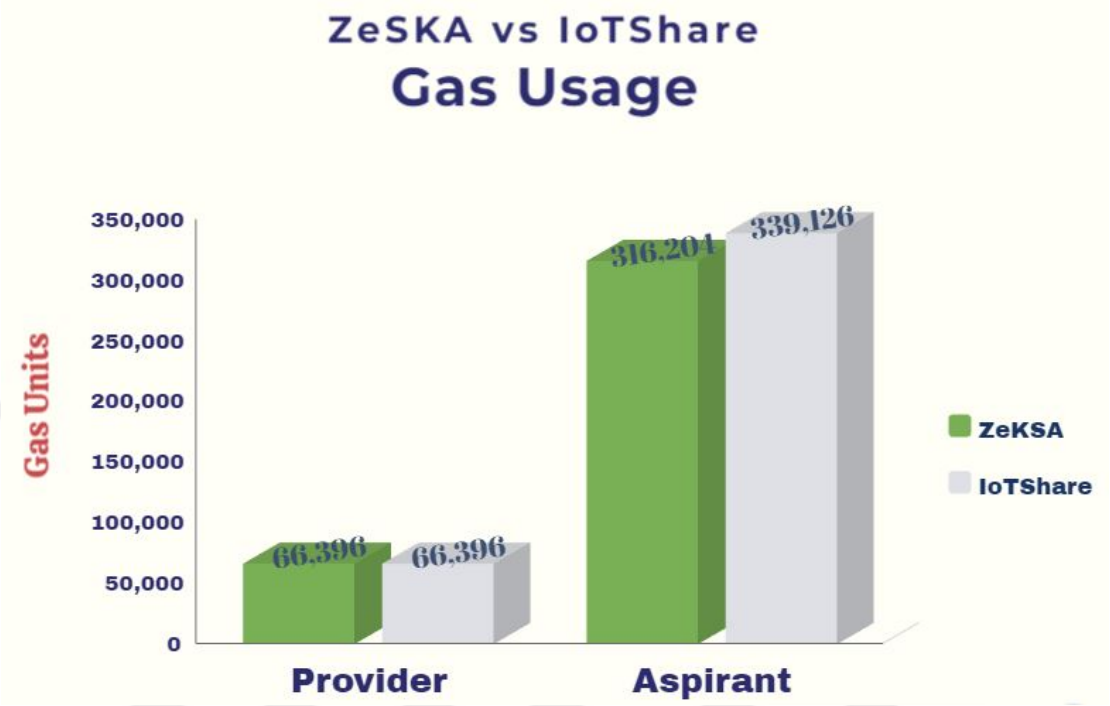

Figure 8: Gas usage Comparison

ZeKSA, on average consumes, 66395 Gas as seen in figure 8. However, there is an interesting observation to make when compared to the IoTShare approach. The aforesaid non-zk approach consumes almost the same amount of Gas as ZeKSA, with a minuscule difference being that the non-zk approach is taking up 0.003% more Gas. This similarity can be attributed to the fact that, most of the work done by provider devices actually occurs outside the blockchain and subsequently outside the network itself. The majority of their time is spent in monitoring the network for suitable resource requests, which is synonymous to an inexpensive read operation in computing. As such, the provider devices do not incur much Gas usage and are usually passive on the network. This explains the logic behind the Gas usage for providers being practically the same on both approaches.

However, the outlook is visibly different when we compare the aspirant devices.

[5]https://goerli.etherscan.io/

Since aspirant devices are active participants on the blockchain, it is mandatory that they use up more Gas. As shows in 8, the ZeKSA version, on average, uses 316204 Gas. In contrast, the IoTShare approach uses 339126 Gas. This translates to an increase of over 6.76% when compared to it's zk-secure counterpart. It should be noted that the IoTShare version of the aspirant device simply issues payments through a public transaction on the provider interface. In comparison, the ZeKSA aspirant device, when making privacy-preserved payments to a provider, invokes the zk-module which utilizes the computationally expensive SNARKs (Succinct Non-interactive Argument of Knowledge) [40] to generate a zero-knowledge secured note. So, despite having a computationally straightforward workflow, the IoTShare approach consumes more Gas and as a result has a more expensive & resource-hungry workflow when compared to it's more secure counterpart.

4.3.3.3 Transaction cost

In order to test the feasibility of operating ZeKSA on the Ethereum mainnet in due course, we also observed the transaction costs incurred by the procedure calls that were invoked during the protocol. These costs are usually calculated by multiplying the gas units used in the transaction to the gas price that was specified by the testnet at the time of the transaction. Transaction costs are usually given in the denomination of Gwei, where 1 ether equates to a billion Gwei units ($1 * 10^9$). To ensure a fair evaluation, procedure calls of the competing protocol were also measured for comparison purposes.

Firstly, the costs between the ZeKSA provider and IoTShare provider device are analyzed. As shown in Figure 9 the ZeKSA provider device, on average, consumes around 535000 Gwei for one complete resource sharing cycle. In contrast, the competing protocol's provider device incurs about 569000 Gwei. This indicates a difference of about 6 %, which is quite significant considering the present day fiat value of ethereum's native cryptocurrency itself[6].

[6]https://www.binance.com/en/price/ethereum

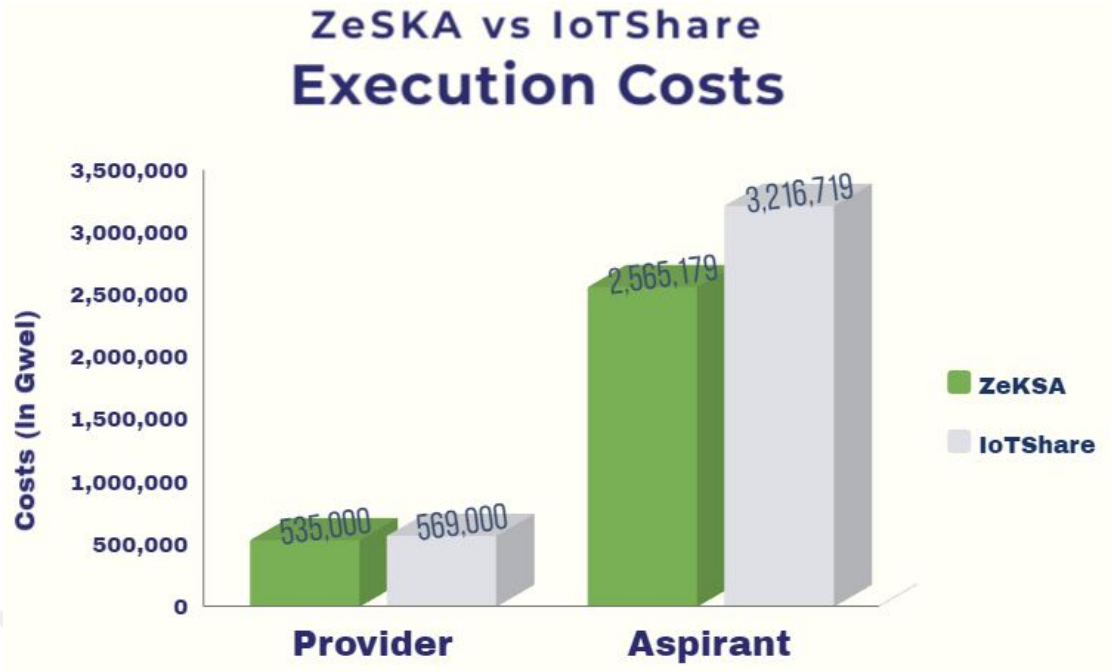

Figure 9: Execution cost comparison

On the other hand, aspirant devices, which constitute a more hands-on presence on the blockchain & subsequently incur higher costs naturally. In spite of this, ZeKSA appears to be more efficient as its aspirant consumes around 2565179 Gwei for one complete resource sharing workflow. While this might seem high compared, it is still cheaper than what the IoTShare provider demands. The competing protocol's aspirant device consumes around 3216719 Gwei for one complete workflow. This translates into a 20.25% increase when compared to the ZeKSA scheme and it is indeed quite a substantial amount. Despite having a seemingly straight forward interface for exchanging resources and issuing publicly provable payments, the competing protocol appears to be more expensive in terms of transaction cost when compared with ZeKSA.

After going through the empirical analysis of the results, we can conclude that despite being quite computationally sophisticated and possessing a highly meticulous mode of operation, evaluation results indicate that ZeKSA is more efficient in terms of time and resource consumption while also being sensitive to transaction costs, user security & privacy.

4.4 Security Consideration

In this segment, we evaluate the security of the resource sharing mechanism deployed of the proposed research. We achieve this by presenting an exhaustive discussion pertaining to the likelihood of an adversary encroaching upon the security of the proposed mechanism by trying to intercept the exchange of critical data between the aspirant and the provider device.

Firstly, we elaborate the security assumptions

4.4.1 Security Assumptions

Prior to conducting a security analysis of the system, we clearly underline the security assumptions that we hold for the system.

The analysis will hold the following assumptions :

1. Each device is given a unique public/private key pair (P_{bkx}, P_{vkx}) upon being registered on the network. The public key P_{bkx} is openly broadcast for communication & verification purposes while the private key remains secret, known only to the respective owner device, whom, it was originally assigned to e.g for the aspirant IoT device I_a, it owns the keypair (P_{bka}, P_{vka}) and a provider IoT device I_p it owns the keypair (P_{bkp}, P_{vkp})

2. The asymmetric encryption scheme used by the participants (IoT devices) is contingent upon the security of the respective private keys, i.e., it is not possible to decrypt a data encrypted by public key of provider device P_{bkb} without having access to the private key of the provider device P_{vkb}.

We achieve this through Public Key Encryption, which is described as follows.

4.4.2 Public Key Encryption

Public key cryptography uses a pair of keys: a public key and a private key. The public key is used to encrypt messages, while the private key is used to decrypt them. The notation for this, in terms of the proposed work, is represented as follows:

$Publickeys : (P_{\text{bka}}, P_{\text{bkp}})$

$Privatekeys : (P_{\text{vka}}, P_{\text{vkp}})$

Where P_{bka} and P_{bkp} are the public keys for aspirant and provider respective. Meanwhile, P_{vka} and P_{vkp} private keys, for aspirant and provider. The public and private keys are related in such a way that messages encrypted with the public key can only be decrypted with the corresponding private key.

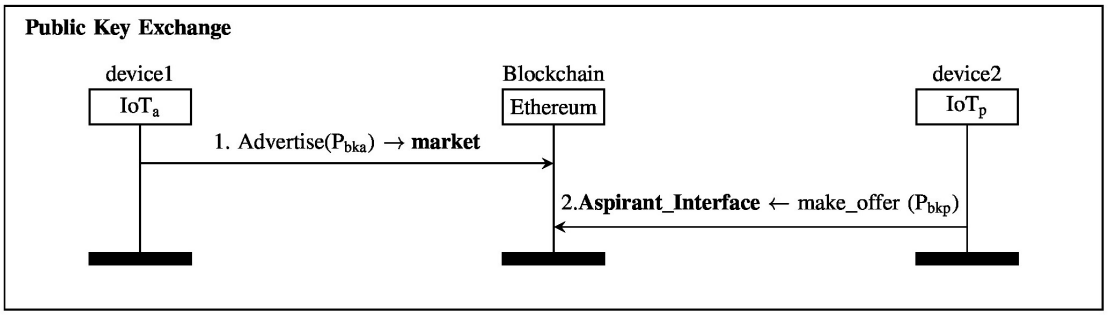

Figure 10: Public Key Exchange

Foremost, we look at the Public Key exchange procedure, which is quite subtle and passive in nature. According to Figure 10, by advertising a resource request through the market contract in message **1**, an aspirant device IoT_a effectively shares its public key P_{bka} with any prospective provider device that is able to listen. Conversely, in the same illustration, by making an offer (message **2**) through the interface of a particular aspirant, a provider IoT_p also shares its public key, P_{bkp}, albeit with only that particular aspirant. The pattern of this exchange is coherent with the protocol's principle of operation, since provider's offer is based on the concept of a one-to-one communication, while an aspirant's resource request advertisement is reminiscent of a one-to-community communication [41]. This exchange might seem inadvertent, but is actually essential in the event that a resource sharing understanding is reached between the devices in the future.

We briefly look at how ZeKSA allows for the relaying of sensitive information, like workloads & redeemable zk-notes meant to be specifically private to a provider device, through a publicly accessible smart contract. To concretely understand this premise, it is essential to firstly observe how the devices broadcast their public keys

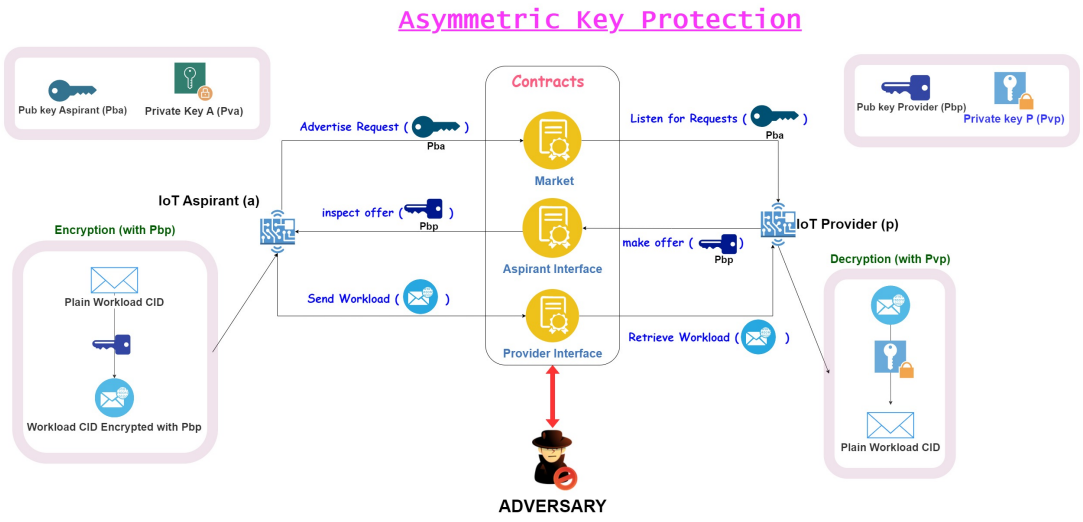

Figure 11: Asymmetric Key Encryption

with the help of blockchain. In Figure 11, the adversary model is explained with one of the possible scenarios. Prior to this, it is essential to look at the Figure 6.

We briefly discussed in section 3.3 that workload content and zk-secure payments are shared between the aspirant & provider devices through the provider interface. This interface is a smart contract which is publicly accessible on the blockchain & any other device with knowledge of the provider interface contract address can try to make adversarial procedure calls. Yet, despite this, the workload, its output and the payment are all safely delivered to the intended recipients. This is possible because of asymmetric encryption, as shown in Figure 11.

To explain the security of sharing, we take the example of an aspirant device $\mathrm{IoT_a}$ trying to send the workload to a provider $\mathrm{IoT_p}$. We already know from prior discussion that both devices are aware of each other's public keys. Therefore, to ensure security of it's workload, the aspirant device encrypts it workload CID using the public key $\mathrm{P_{bp}}$ of provider device. Now, as per the principles of asymmetric key cryptography, the encrypted payload can only be viewed by an entity that holds the private key corresponding to $\mathrm{P_{bp}}$. When the aspirant device tries to send its encrypted payload to the provider device through the Provider Interface, there is a risk that an adversary might also be present on the blockchain. This

adversary will try to intercept the information. Since ethereum is fundamentally a permissionless ledger, potentially anyone can join the network and start interacting with the on-chain code. However, even in the circumstance that the adversary can successfully intercept a data exchange between a provider and an aspirant, it will only be able to steal encrypted information. This information, which has been encrypted by the public key of the provider device, P_{bp}, will remain in obfuscated form until the appropriate coresponding private key is provided. That private key is P_{vp} and it belongs to IoT_p. Therefore, in this situation only IoT_p can decrypt and see the data in it plain form. Any other network participant, cannot see the unencrypted form of the data. Hence, when the provider is eventually able to retrieve this encrypted ID from the Provider Interface, it uses its own private key and successfully decrypts the content ID. The content ID can be used to retrieve the actual workload from IPFS. Eventually, the output of the workload and the zk-secure note, which are being sent over the blockchain network in messages **13** and **18** of Figure 6 respectively, are also secured using the same technique. All this is being performed under an optimal and privacy preserved environment; since (1) the third parties have no clue that how much will the provider earn for his services as his payment note is secured through zero-knowledge and also through asymmetric encryption, while also (2) both the participating IoT devices are basically talking to the blockchain and have very limited identifying information (public keys) on each other, so even inter-device privacy is being secured within ZeKSA's framework.

4.4.3 Security Analysis

ZeKSA offers guarantees for device anonymity and transaction amounts which are paid to provider devices in lieu of their role in resource sharing. Since the transaction obfuscation logic is based on Tornado Cash (TC), therefore the threat model will be presented accordingly.

As discussed in [42], decentralized mixing paired with Zero-Knowledge proofs makes for a strong transaction privacy apparatus. This guarantee, however, is

contingent on the strength of the anonymity set formed by TC. Thus anonymity is the asset of a possible threat model. The Potential threat is the de-anonymization of the amount of payment being made.

Anonymity sets were first suggested by David Chaum [43] in 1981, through a technique which is sometimes referred to as the Chaumian Coin Toss. This qualifies as one of the earliest examples of an anonymity set, which can be defined as a group of elements used to obscure the identity of individual users or transactions within the group. Recently, anonymity sets have become popular within the context of blockchain. Tornado Cash uses zero-knowledge proofs in tandem with other techniques to create large anonymity sets, therefore making it outrageously laborious to determine the exact source of specific transactions within the set.

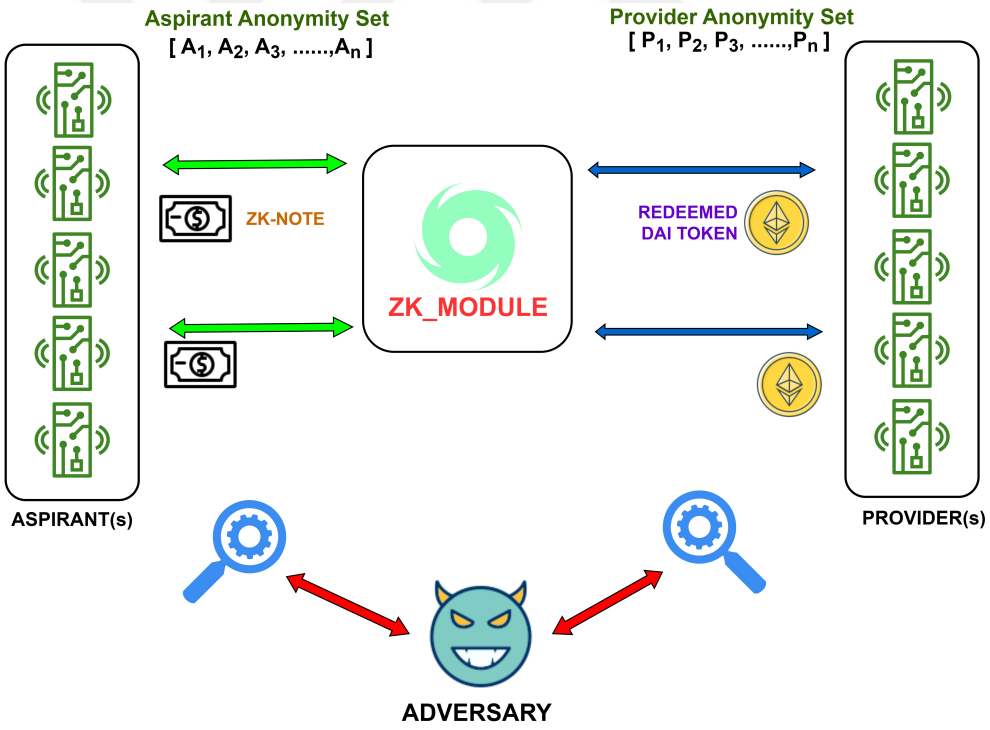

Figure 12: Threat Model

In Figure 12, an adversary (A) can be seen trying to attack the ZeKSA system. A is basically trying to monitor the system and de-anonymize Zero-Knowledge transactions happening between Aspirant and Provider devices. As previously explained in Section 4.4.2, it is not possible for a third party to decrypt an encrypted

ZK-Note during transmission from Aspirant to Provider. Therefore, the adversary cannot intercept a note to determine it's worth and cannot figure out the transaction amount. Thus, public key encryption is the first mitigation measure in this threat model.

Moreover, once again referring to Figure 12, A might try to deanonymize payment amounts between specific Aspirants and Providers. For instance, it might try to find out exactly how much did Aspirant (A_1) paid to a specific Provider (P_3). This would be a threat to the privacy-preserved payment facility of ZeKSA.

In the scenario where if there was only one depositing Aspirant device and a single Provider device making a withdrawal, it would be clearly obvious to link the payment amount between the two devices. However, for a sufficiently large anonymity set of Aspirants depositing to the ZK-module;

$$\text{Aspirant anonymity set} = \{A_1, A_2, \ldots, A_n\}$$

and a sufficiently large anonymity set of Providers withdrawing from the same ZK-module;

$$\text{Provider anonymity set} = \{P_1, P_2, \ldots, P_n\}$$

It is computationally unfeasible for Adversary A to try and figure out the amount of payment exchanged between A_1 and P_3, without knowing the secret contents of the ZK-note that was shared between them. Since several Aspirants will be depositing to the ZK-module and several Providers will be withdrawing from the same module, with zero-knowledge proofs breaking the link between the depositing and withdrawing entities, it will be extremely arduous for A to know that exactly which withdrawal corresponds to which deposit. Therefore anonymity sets act as the second security measure for this threat model and it consequently mitigates the threat of patternizing and deanonymizing private payments between IoT devices which would be transacting through ZeKSA.

CHAPTER V

CONCLUSION

In this research, we explore paradigm of decentralized resource sharing between power constrained and limited-capability IoT devices. ZeKSA is presented as a framework to ensure reliable and privacy-preserved resource sharing between devices tethered to a public ledger. As a first time innovation, we also combine zero-knowledge proofs to the resource sharing protocol by integrating the ethereum-based Tornado Cash as a medium of privacy-preserved payment. Further security is guaranteed through asymmetric key cryptography for encrypt sensitive exchanges between interacting devices. An additional privacy-preserving measure is presented through ensuring that intermingling devices maintain extremely stringent anonymity between themselves by sharing only their public keys with potential aspirant/providers. Other than that, the devices have no knowledge about other network participants and this guarantees absolutely anonymous resource sharing. Experimental evaluations focused on pure DLT metrics like gas usage and transaction costs yielded compelling outcomes indicating that ZeKSA is both, computationally & operationally feasible in addition to being sensitive with user privacy. Possible future work can involve Additionally, real-time testing of this framework by embedding it within an ecosystem of physical IoT devices. Additionally, evaluating it with a regulated DLT network, like r3 Corda [44] might make for an interesting and insightful future analysis. Moreover, expanding this protocol to macro implementations, for use with several scenarios like Industrial IoT (IIoT) [45] also constitutes a promising prospect for further experimentation.

REFERENCES

[1] L. G. Roberts and B. D. Wessler, "Computer network development to achieve resource sharing," (New York, NY, USA), Association for Computing Machinery, 1970.

[2] D. C. Walden, "Systems for interprocess communication in a resource sharing computer network," *RFC*, vol. 62, pp. 1–20, 1970.

[3] L. Roberts, "The arpanet and computer networks," (New York, NY, USA), Association for Computing Machinery, 1986.

[4] K. Ashton, "That 'internet of things' thing," *RFID Journal*, vol. 22, no. 7, pp. 97–114, 2009.

[5] E. Foundation, "Geth light client documentation." Available at https://geth.ethereum.org/docs/interface/les. (accessed 13 December, 2022).

[6] J. Benet, "IPFS - content addressed, versioned, P2P file system," *CoRR*, vol. abs/1407.3561, 2014.

[7] P. Voigt and A. v. d. Bussche, *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer Publishing Company, Incorporated, 1st ed., 2017.

[8] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on Computing*, vol. 18, no. 1, pp. 186–208, 1989.

[9] N. Szabo, "Formalizing and securing relationships on public networks," *First monday*, 1997.

[10] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[11] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for iot security and privacy: The case study of a smart home," in *IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pp. 618–623, 2017.

[12] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[13] N. Aleisa and K. Renaud, "Privacy of the internet of things: a systematic literature review (extended discussion)," *arXiv preprint arXiv:1611.03340*, 2016.

[14] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, pp. 2–1, 2014.

[15] C. Team, "Crypto mixers and aml compliance," Aug 2022.

[16] T. Bontekoe, "Balancing privacy and accountability in digital payment methods using zk-snarks.," October 2020.

[17] J. W. Palfner, "Tornado vote: secure and private e-voting based on tornado cash," Master's thesis, Universitat Politècnica de Catalunya, 2021.

[18] D. K. Sharma, N. C. Singh, D. A. Noola, A. N. Doss, and J. Sivakumar, "A review on various cryptographic techniques & algorithms," *Materials Today: Proceedings*, vol. 51, pp. 104–109, 2022.

[19] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Scalable, transparent, and post-quantum secure computational integrity," *Cryptology ePrint Archive*, 2018.

[20] P. L. G. Ramírez, M. Taha, J. Lloret, and J. Tomás, "An intelligent algorithm for resource sharing and self-management of wireless-iot-gateway," *IEEE Access*, vol. 8, pp. 3159–3170, 2020.

[21] I. U. Din, K. A. Awan, A. Almogren, and B.-S. Kim, "Sharetrust: Centralized trust management mechanism for trustworthy resource sharing in industrial internet of things," *Computers and Electrical Engineering*, vol. 100, p. 108013, 2022.

[22] R. Gupta, I. Gupta, A. K. Singh, D. Saxena, and C.-N. Lee, "An iot-centric data protection method for preserving security and privacy in cloud," *IEEE Systems Journal*, pp. 1–10, 2022.

[23] A. Aktypi, K. Kalkan, and K. B. Rasmussen, "Secas: Secure capability sharing framework for iot devices in a structured p2p network," in *Proceedings of the Tenth ACM Conference on Data and Application Security and Privacy*, pp. 271–282, 2020.

[24] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," vol. 31, no. 4, 2001.

[25] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *IFIP/ACM International Conference on Distributed Systems Platforms and Open Distributed Processing*, pp. 329–350, Springer, 2001.

[26] M. Conoscenti, A. Vetrò, and J. C. De Martin, "Peer to peer for privacy and decentralization in the internet of things," in *IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pp. 288–290, 2017.

[27] Z. Wilcox-O'Hearn and B. Warner, "Tahoe: the least-authority filesystem," in *Proceedings of the 4th ACM international workshop on Storage security and survivability*, pp. 21–26, 2008.

[28] T.-H. Hsu and Y.-M. Tung, "A social-aware p2p video transmission strategy for multimedia iot devices," *IEEE Access*, vol. 8, pp. 95574–95584, 2020.

[29] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[30] B. Hamdaoui, M. Alkalbani, A. Rayes, and N. Zorba, "Iotshare: A blockchain-enabled iot resource sharing on-demand protocol for smart city situation-awareness applications," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10548–10561, 2020.

[31] S. Iqbal, R. M. Noor, A. W. Malik, and A. U. Rahman, "Blockchain-enabled adaptive learning-based resource sharing framework for iiot environment," *IEEE Internet of Things Journal*, 2021.

[32] S. Biswas, K. Sharif, F. Li, B. Nour, and Y. Wang, "A scalable blockchain framework for secure transactions in iot," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4650–4659, 2019.

[33] L. Shi, X. Li, Z. Gao, P. Duan, N. Liu, and H. Chen, "Worm computing: A blockchain-based resource sharing and cybersecurity framework," *Journal of Network and Computer Applications*, vol. 185, p. 103081, 2021.

[34] Ethereum, "Go ethereum client (geth)." Available at https://github.com/ethereum/go-ethereum, 2015. (accessed 13 December, 2022).

[35] N. Alsalami and B. Zhang, "Sok: A systematic study of anonymity in cryptocurrencies," in *IEEE Conference on Dependable and Secure Computing (DSC)*, pp. 1–9, 2019.

[36] D. Khovratovich and M. Vladimirov, "Tornado privacy solution: Cryptographic review. version 1.1," 2019.

[37] D. Boneh, A. Gervais, A. Miller, C. Parlour, and D. Song, "Decentralized finance - privacy on the blockchain," Oct 2021.

[38] Y. Tang, C. Xu, C. Zhang, Y. Wu, and L. Zhu, "Analysis of address linkability in tornado cash on ethereum," in *China Cyber Security Annual Conference*, pp. 39–50, Springer, 2021.

[39] S. Prime, "Lighthouse." Available at https://github.com/sigp/lighthouse, 2018. (accessed 13 December, 2022).

[40] N. Bitansky, R. Canetti, A. Chiesa, and E. Tromer, "From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again," (New York, NY, USA), Association for Computing Machinery, 2012.

[41] J. Fawkes and A. Gregory, "Applying communication theories to the internet," *Journal of communication management*, 2000.

[42] M. C. Kus Khalilov and A. Levi, "A survey on anonymity and privacy in bitcoin-like digital cash systems," *IEEE Communications Surveys Tutorials*, vol. 20, no. 3, pp. 2543–2585, 2018.

[43] D. Chaum, "Security without identification: Transaction systems to make big brother obsolete," *Commun. ACM*, oct 1985.

[44] M. Hearn and R. G. Brown, "Corda: A distributed ledger," *Corda Technical White Paper*, vol. 2016, 2016.

[45] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, "The industrial internet of things (iiot): An analysis framework," *Computers in Industry*, vol. 101, pp. 1–12, 2018.

VITA

Muhammad Yasir is a M.Sc Student working with the supervision of Dr.Kübra Kalkan and is concurrently a Teaching Assistant at the Department of Computer Science, Özyeğin University, Istanbul. He completed his Bachelors Degree in Computer Systems Engineering from NED University in 2017 in Pakistan and has previously worked as a Research Assistant at National Center for Cyber Security, also at NEDUET. His research interests include emerging technologies like blockchain, big data, and artificial intelligence.