Towards Sustainable Blockchains:
Cryptocurrency Treasury and General Decision-making Systems with Provably Secure Delegable Blockchain-based Voting

School of Computing and Communications
Lancaster University

Hamed Olanrewwaju Balogun

Supervisors: Prof. Bingsheng Zhang
Dr Antonios Gouglidis

This thesis is submitted for the degree of
Doctor of Philosophy

July, 2021

To the memory of my late Uncle, Alhaji Adiama Imam Yusuf...

Abstract

The blockchain technology and cryptocurrencies, its most prevalent application, continue to gain acceptance and wide traction in research and practice within academia and the industry because of its promise in decentralised and distributed computing. Notably, the meteoric rise in the value and number of cryptocurrencies since the creation of Bitcoin in 2009 have ushered in newer innovations and interventions that addressed some of the prominent issues that affect these platforms. Despite the increased privacy, security, scalability, and energy-saving capabilities of new consensus protocols in newer systems, the development and management of blockchains, mostly, do not reflect the decentralisation principle despite blockchains being decentralised and distributed in their architecture. The concept of treasury has been identified as a tool to address this problem. We explore the idea of blockchain treasury systems within literature and practice, especially with relation to funding and decision-making power towards blockchain development and maintenance. Consequently, we propose a taxonomy for treasury models within cryptocurrencies. Thereafter, we propose an efficient community-controlled and decentralised collaborative decision-making mechanism to support the development and management of blockchains. Our proposed system incentivises participants and is proven secure under the universally composable (UC) framework while also addressing gaps identified from our investigation of prior systems e.g. non-private ballots and insecure voting. Furthermore, we adapt our system and propose a privacy-preserving general decision making system for blockchain governance that supports privacy-centric cryptocurrencies. Besides, using a set of metrics, we introduce a consensus analysis mechanism to enhance the utility of decision-making of the systems by evaluating individual choices against collective (system-wide) decisions. Finally, we provide pilot system implementations with benchmark results confirming the efficiency and practicality of our constructions.

Declaration

I hereby declare that except where specific reference is made to the work others, the contents of this thesis are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or at any other university. This thesis is my own work and outcomes of work done in collaboration with others is specified in the text and Acknowledgement.

Hamed Olanrewaju Balogun

September, 2020

Acknowledgement

My debt of gratitude goes out to my supervisors Dr Bingsheng Zhang and Dr Antonios Gouglidis for their immense guidance, patience and effort towards the success of my PhD program. I am eternally grateful for all of the kindness and advice and support I have received over the course of my study.

To Dr Bingsheng Zhang, I say a big thank you! I had the immense pleasure of listening to, learning from, and working with you. I thank you for the invaluable experience of your mentorship, support and time without which it would be impossible to complete this PhD program. I look forward to continue working with and learning from you. Thank you very much.

To Dr Antonios Gouglidis who co-supervised my thesis, I say a big thank you for the invaluable guidance and thorough revision of this thesis and fruitful criticism that has helped improve the overall quality of the output. I am privileged to have had the opportunity of working under your stewardship. Thank you once again for the support, guidance and encouragement. I look forward to continue working with and learning from you.

To Dr Jose Such, my initial supervisor, I say a big thank you for the opportunity to work with you, the belief, support and encouragement you have provided me through out my career (from my masters). I will forever be grateful.

To my examiners Prof. Joe Finney and Dr Cong Wang, I say a big thank you for your time, effort and insightful comments.

I would like to specifically thank and appreciate my love Kamilah Aliyu and my prince Aliy 'Baby' Balogun for their immense patience, and help towards the completion

of this thesis. This would have been impossible without your love and support that have helped me push through.

I would also like to thank Dmytro Kaidalov and Andrii Nastenko (IOHK) for the intellectual discussions and the Scala prototype implementations and benchmarks.

I am also hugely indebted to all of my teachers from my formative years whose kindness, support and encouragement have been crucial throughout this journey. Specifically, I would like to thank Mr J.S. Ajayi (ABS), Mrs Umar (Federal Government College, Ilorin), Dr A.O. Otuoze, Dr J.F. Opadiji, Prof. Y.A. Adediran (University of Ilorin) and Prof Awais Rahid (Lancaster University).

I am also grateful to my panel members for their efforts and guidance which has helped shape the direction of my research. Special gratitude also goes to all the administrative staff at the School of Computing and Communications who have contributed in one way or another to ensuring my experience at the department has been nothing but wonderful.

I would also like to thank all of my friends and colleagues: Jide Edu, Farid Bello, Engr. Samuel Onidare, Nasser Al-Salami, Peng Cheng, Col. Anthony Mazeli, Mr Kabir Balogun, Dr Wasiu Balogun, Udoh Itoro, Zenpeng Li, Mei Wang, Jiajie Zhang, Dr Marvin Ramokapane, Dr Ethem Bagci, Dr Gaurav Misra, Pierre Ciholas.

I reserve a special thank you to Abubakar Jibrin (Jada), Dr Muhammad Kabir Salihu and Aisha Iya Abubakar for their innumerable support, friendship, and convenient distraction and useful motivation they have provided me from the first day of my PhD program. I am eternally grateful to you guys.

This thesis would not have been possible without the care and support of my loving parents, Alh. I. O. Balogun, Mrs T. Balogun and Mrs N. Balogun. I am also eternally grateful to my siblings: Aisha, Rafiat, Wasiu, Shukurat, Monsurat, Sis Shakirah, Bro Musbau, Bro Afeez, Sis Rashidah, and Bro Taofeek; and family members: Engr. and Mrs Mohammed, Uthman, Mubarak and Yahaya, my aunt Alhaja B. Mohammed, Alhaja Aliyu and the girls (Aisha, Zeenat, Asmau, and Sarah), and Alhaji Akolade for their love, care, support and prayers. To all my siblings, I say a big thank you for your love and support. I am grateful to have the best siblings in the world. I love you all. Thank

you to the rest of my family and friends.

Finally, I am grateful to the Almighty Allah for His abundant blessings upon me, my family and friends.

Publications

Part of the research documented in this thesis appeared in workshops/conferences. Below is a list of publications from some of the research presented in this thesis.

- **Hamed Balogun**, Bingsheng Zhang, and Roman Oliynykov: "Privacy-preserving Collaborative Decision-making for Blockchain Governance." (Journal Submission Under Review)

- Bingsheng Zhang, Roman Oliynykov, **Hamed Balogun**: "A Treasury System for Cryptocurrencies: Enabling Better Collaborative Intelligence." *Proceedings of the 26th Annual Network and Distributed System Security Symposium.* NDSS 2019.

- **Hamed Balogun**, Bingsheng Zhang: "On the Sustainability of Blockchain Funding." *Proceedings of the 2018 IEEE International Conference on Data Mining Workshops, ICDM Workshops.* IEEE 2018: 89-96

- Bingsheng Zhang, Roman Oliynykov, **Hamed Balogun**: " A Treasury System for Cryptocurrencies: Enabling Better Collaborative Intelligence." *International Association for Cryptologic Research Cryptology ePrint Archive.* IACR 2018: 435

Other publications orthogonal to this thesis are listed below.

- Gaurav Misra, Jose M. Such, **Hamed Balogun**: "Non-sharing communities? An empirical study of community detection for access control decisions." *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining.* ASONAM 2016: 49-56

- Gaurav Misra, Jose M. Such, **Hamed Balogun**:“ IMPROVE - Identifying Minimal PROfile VEctors for Similarity Based Access Control.” *Proceedings of The 15th IEEE International Conference on Trust, Security, and Privacy in Computing and Communications.* Trustcom/BigDataSE/ISPA 2016: 868-875

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Since the invention of the pioneer blockchain and cryptocurrency Bitcoin [Nak08] in 2009, there has been a remarkable increase in the number and value of cryptocurrencies, as well as blockchain research [Blo]. Similarly, there have been an increased acceptance, adoption and application of blockchains and cryptocurrencies across various domains and industries e.g. finance, insurance, healthcare, social computing, governance, research, Internet of Things [ZS18, CXS+19, CDP19, MCLH19, WZN+19, XTH+19]. The distributed and decentralised architecture of blockchains have enabled the construction of novel systems that do not depend on trusted third parties across different sectors.

Cryptocurrency is the foremost application of blockchains [BMC+15, MSH+16, NBF+16, LXS+19, Zca, DASa, VVL17]. Cryptocurrencies as decentralised ledgers, enable payment transactions among parties, that are, perhaps, mutually distrusting, pseudonymous or anonymous. According to Coinmarketcap.com, a foremost cryptocurrencies data platform, there are over 700 cryptocurrencies in existence, with a combined market capitalisation of over $1 trillion [Coi]. This growth have necessitated wide-ranging improvements, regulations, legislations, user-interaction, engagement, community participation and novel blockchain technologies that improve on early constructions such as Bitcoin. For instance, newer protocols with strong security and privacy guaran-

tees [KKKZ19, HBHW17, GHM+17], as well as distinctive underlying cryptographic constructions such as cryptonote [vS13] and tangle [PSF19].

Other improvements include: development of ASIC-resistant mining puzzles through memory-bound and memory-hard puzzles, e.g. Scrypt [PJ16]; more privacy-centric cryptocurrencies e.g. Zcash [Zca], Monero [Mon]; new variants of proof-of-work consensus e.g. Proof-Activity [BLMR14]; and new blockchain consensus protocols, e.g. Proof-of-stake [GHM+17], Ouroboros [KRDO17]. Other developments in blockchain research and practice have resulted in the creation of smart contract platforms, e.g. Ethereum [Woo14] and EOS [1], which are able to execute 'legal agreements' that are implemented in computer programs. This represents an upgrade from previous blockchain platforms that offered mainly basic payment transactions and limited computational capabilities [AO13] such as notary services [2].

Cryptocurrencies have evolved from first generation blockchains that mainly facilitate payment transactions among different parties e.g. Bitcoin, Litecoin [Tak18] and support limited computational tasks to second generation blockchains that provide self-enforcing agreements in programmable logic (smart contracts), that digitally mimic real world contracts e.g. Ethereum [Woo14, ASB+17, MMAM+16, KMS+16]. Meanwhile, third generation blockchains mainly seek to address scalability, sustainability and interoperability among different blockchains [MBB+19, KJG+18, ZMR18, YWY+20]. Some solutions proposed to improve scalability include the use of off-chains for separation of standard payment transactions (parallel payment networks) from main blockchain (core) [PD16, MMK+17, GM17, KG17]; side-chains [BCD+14, CDE+16, GKZ19]; and sharding [KJG+18, ZMR18, MBB+19, WSNH19, YWY+20]. Solutions for inter-operability among various blockchain technologies include [ZCC+16a, BCD+14, LXS+19, ASB+17, Her18, MMAM+16, KZ19, GKZ19].

Key sustainability issues are environmental, ethical and governance of the operations of blockchains [Vra17, GRTT17]. For example, Bitcoin with an estimated electricity con-

[1] https://eos.io/

[2] blocknotary.com, notarised.io, stampd.io, notary.bitcoin.com

sumption of 68.35TWh [3] consumes more energy than some countries such as Switzerland, Ireland, Kuwait, and Czech Republic, according to the International Energy Agency. Consequently, alternative consensus protocols such as proof-of-stake have been proposed and adopted by newer blockchains and cryptocurrencies [4] [GHM+17, KRDO17, KKKZ19]. Moreover, older cryptocurrencies such as Ethereum are also migrating from proof-of-work consensus to proof-of-stake consensus protocols to address the problem of excessive energy usage [5].

Despite improved constructions, scalability, interoperability, privacy and security in blockchains, the operation and maintenance of these platforms do not reflect the decentralisation ethos embodied at the protocol level (layer), i.e. development and management operations have remained largely centralised. In other words, the governance by infrastructure is decentralised, while development and maintenance of the infrastructure is not decentralised. This observation is fundamental to the contributions of this thesis.

Previous research works have investigated and documented centralisation risks (of governance structures) within blockchains [GKCC14, MC13]. For instance, by investigating the decision-making process on the issue of increasing the block size, Filippi et. al, [DFL16] explained that Bitcoin's maintenance and development depends largely on a small core of highly skilled developers. Similarly, Azouvi et. al. [AMM18] reveal only a handful of users account for the majority of discussions and code towards cryptocurrency development in Bitcoin and Ethereum. Clearly, developers and/or founders hold significant amount of power, and therefore can effectively influence decision-making processes, which in turn, can lead to unhealthy centralisation within the system.

To mitigate centralisation risks in blockchain development and maintenance, some cryptocurrencies such as Dash [DASa, Dasb], Zcash [Zca, Fou], and Decred [Deca, Decb] provide periodic funding to support projects that advance their platforms or blockchain development in general. Examples of projects include: marketing activities, research, development of blockchain tools (e.g. wallets), etc. Typically, this funding is provided

[3] www.cbeci.org
[4] A list of proof-of-stake coins can be found at cryptoslate.com/cryptos/proof-of-stake/
[5] https://ethereum.org/en/eth2/

to any stakeholder or user or group interested in carrying out projects that support the development of the blockchain. Usually, decision-making on fund distribution to projects is done through voting, involving only a select group of participants or centrally decided by an appointed board or committee. Participation in the process is not incentivised (low-turnout) and decision-making processes are not securely coupled with the blockchain and basic voting systems for determining winning projects are supported.

The main aim of this research is to provide provably secure and practical decentralised system that support blockchains towards sustainable development and maintenance. Specifically, we propose a novel cryptocurrency treasury system to support decentralised utilisation of blockchain resources, by all stakeholders, for the advancement of the blockchain. Mainly, the treasury system covers the sourcing of funds for proposals, proposal submission process, decision-making on how funds are utilised and the eventual distribution of funds to successful projects. The proposed treasury system is mostly platform agnostic and can support cryptocurrencies with payment transactions. Moreover, we build upon our treasury system and propose a privacy-preserving general decision-making system for cryptocurrencies with private stake information. We remark that, a key requirement of any proposed solution is that it should not introduce additional centralisation to the blockchain as a result of its operation.

Therefore, to achieve the research aim, we outline the following pertinent objectives. Primarily, this research investigates different sources of funding blockchain development by exploring funding in cryptocurrencies. The examination resulted in identification, classification of cryptocurrency funding and their associated potential effects on decision-making within blockchains. Consequently, solutions that ultimately limit unhealthy influence of powerful minority within blockchains, and better aggregate the collective input of members of the cryptocurrency ecosystem were proposed (for adoption in the designed treasury system).

Furthermore, to mitigate centralisation tendencies and powerful minorities in the development and maintenance of blockchains, this work explored the degree of involvement of holders of stake in a cryptocurrency in decision-making processes for maintaining

and advancing the cryptocurrencies. Hence, this research combined insights identified from the exploration of cryptocurrency funding mechanisms and decision-making, and successfully proposed practical treasury system for cryptocurrencies. The treasury system combines novel ideas from collaborative intelligence gathering (liquid democracy), decision-making processes, and new efficient zero-knowledge constructions. The proposed treasury solution has been adopted by real-world cryptocurrency platforms such as Horizon [Hor18] (formerly Zencash) and Cardano [Car18b](IOHK).

To guarantee increased (stake-)participation in the treasury and decision-making systems, we abstracted a cryptocurrency coin into the notion of coin-ownership and stake-ownership. Essentially, the owner of a coin is the only user that can successfully spend a coin while a stake-owner is any user who possesses the stake/value of a coin for the purpose of decision-making or participating in the treasury system. Therefore, the system support permanent/offline stake delegation, and also have an additional benefit of potentially increasing the total amount of stake (participation) in any treasury/decision-making period. This is because coin owners do not need to be online for their stake to be involved in the treasury decision-making process.

Our proposed treasury system mainly applies to blockchains with public-stake information. However, there is an increasing number of (recent) blockchains where stake information is private, i.e. cryptocurrencies where the amount of stake owned by any user/stakeholder is not publicly available on the distributed ledger. Therefore, to address this potential limitation of the application of the treasury system, we develop a general collaborative decision-making system for blockchains with private stake. This research work also proposed an efficient privacy-preserving general decision-making system for blockchains using novel multiplicative-relation zero-knowledge proofs. Particularly, the system is compatible with cryptocurrencies with hidden/private stake information e.g. Zencash [BCG+14, vS13, KKKZ19].

Finally, to ensure that quality decisions are reached on these systems, the research work also investigates the utility of decisions by analysing soft-consensus among participants in the blockchain decision-making outcomes. Evaluation of soft-consensus assesses agreement

on conclusions reached within the blockchain and provides insights for useful discussions, which in turn, increases community engagement, participation and acceptance of system-wide decisions.

1.2 Problem Statement

Given the established adoption of blockchains across various industries [MA18, AVP+18, PSSK20] and increasing ubiquity of cryptocurrencies in particular, due to their distributed and decentralised architecture, it is pertinent to investigate the degree of decentralisation in the maintenance and development of these platforms. Despite the decentralised architecture, there exists known centralisation threats to blockchains [BS15, AMM18, GBE+18, GKCC14] such as concentration of mining power in mining pools, centralised development, maintenance and governance, etc. Consequently, there have been clear justifications for the blockchain users/community to contribute towards developmental decision-making to address centralisation [But, KKN+17, Decb].

Specifically, we ask the following main research question. *Is it possible to design provably secure blockchain-based self-sustenance mechanisms that will ensure long-term development, maintenance, and sustainability of cryptocurrencies that optimises community contribution without introducing centralisation tendencies?*

Treasury systems have been proposed to address some of these centralisation tendencies in blockchain development and maintenance. Blockchain treasury systems are self-sustenance mechanisms that mainly cover fund sourcing and utilisation for blockchain development. Dash (Dash Governance system [KNS+16]), Decred (Decred treasury [Decb]), and Zcash (Zcash governance [Fou]) are examples of cryptocurrencies that have introduced/adopted a system of treasury, that allows community members contribute to cryptocurrency development, thereby mitigating centralisation in the process of blockchain development directions. Despite this increasing adoption of treasury systems by blockchains, it has not been rigorously studied to ensure secure design and wide adoption and earlier proposals have mostly been ad-hoc solutions with considerable drawbacks. We provide more details about these issues in our exploration of the earlier

proposals in Chapter 2.

For long-term sustainability of cryptocurrencies, the following questions are critical to ensure decentralised development.

- How is the development and maintenance of a cryptocurrency funded?
- Who decides how the funds will be used?

Therefore, we explore the various funding sources for blockchain treasury systems. Specifically, we identify and examine the attributes of the sources of funds for supporting cryptocurrency development, with a view to understanding how they affect the decentralisation and (in turn) the security of their underlying blockchains. Thereafter, we provide a taxonomy of models of treasury system following the exploration of some real-world examples.

Furthermore, early cryptocurrency treasury systems utilise voting systems for allowing users jointly make decisions about developmental proposals for the blockchain. Typically, these voting systems support basic voting schemes, voting is public, not incentivised, and without guarantees or accompanying proofs of security. Additionally, we asked that *can we design and develop a blockchain treasury system with secure voting scheme that helps voters make informed decisions (with minimal effort) and potentially improve on existing solutions? Given the often complex technical nature of some of the proposals being voted on in blockchain treasury systems.*

Consequently, this thesis provides an affirmative answer to the question as demonstrated by the presentations in Chapter 3. Specifically, we designed and developed a secure delegable blockchain-based voting system adopted within our proposed treasury system for blockchains. We also provide details and performance results of a full-prototype implementation of the proposed treasury system.

Moreover, our voting scheme finds application beyond blockchain treasury systems. Hence, we adapt it and proposed a privacy-preserving system for general decision-making on blockchains. The details of this system is provided in Chapter 4. Given the additional potential applications of our secure delegable voting scheme, we explore an avenue

for improving the quality of decisions and acceptance of voting outcomes, by analysing consensus through aggregated distance measurements of voters' choices and system voting outcomes. Chapter 4 provides detailed discussions on the specific distance measures utilised for analysing the consensus on voting outcomes and individual preferences within the system.

In summary, the underlisted **objectives** are key to answering the research questions of this thesis.

- Explore the literature and existing blockchain proposals (for identification of gaps)
- Design of a secure blockchain-based treasury system that optimises community participation through the design of a collaborative decision-making scheme that supports liquid democracy
- Extension of proposed scheme to support privacy-preserving blockchains
- Consensus analysis of system decisions for improved utilisation
- System prototyping and evaluation

1.3 Contributions

The work presented in this thesis explored stable development and inclusive maintenance of blockchains (mostly cryptocurrencies). The goal of this thesis has been to construct provably-secure community-inclusive cryptographic protocols that exploits the decentralised nature of blockchains and optimises community intelligence through appropriate decision-making systems. The major contributions from this thesis are highlighted below.

- **An examination and taxonomy of blockchain (cryptocurrency) development funding, planning, management, and disbursement mechanisms (otherwise known as treasury systems) for cryptocurrencies and blockchain systems.** Chapter 2 of this thesis provides an exploration of blockchain development funding. Specifically, we identified and emphasised the role

of blockchain funding in the overall management of cryptocurrencies and how it is valuable towards long-term cryptocurrency sustainability. In addition to examining the sustainability and highlighting potential issues with various funding sources, we present a taxonomy of blockchain treasury systems based on funding source and method of utilisation. Therefore, we provide a taxonomy of blockchain system reflecting their funding and management system and elaborate on the key features, as well as potential and real issues within each classification. Hence, the closed, open and hybrid treasury system classification.

- **A provably-secure (security) modeling for a blockchain-based treasury voting system that supports delegative voting.** In Chapter 3, for the first time, we model the voting system in the well-known *Universally Composable* (UC) framework [Can00] via an ideal functionality $\mathcal{F}_{\text{VOTE}}^{t,k,n,m}$. The functionality interacts with a set of voters and experts as well as k voting committee members. It allows the voters to either delegate their voting power to some experts or vote directly on the project. If at least t out of k voting committee members are honest, the functionality guarantees protocol termination. Even in the extreme case, when all voting committee members are corrupted, the integrity of the voting outcomes is ensured. However, in that case, protocol termination is not guaranteed.

- **We proposed the world's first honest verifier zero-knowledge proof / argument for unit vector encryption with logarithmic size communication.** Conventionally, to show a vector of lifted ElGamal ciphertexts element-wise encrypt a unit vector, Chaum-Pedersen proofs [CP93] are used to show each of the ciphertexts encrypts either 0 or 1 through Sigma OR composition, and the product of all the ciphertexts encrypts 1. However, the proof size is linear in the length of the unit vector. Therefore, the communication overhead is quite significant when the unit vector length becomes larger. As a result, in Chapter 3 we propose a novel special honest verifier zero-knowledge (SHVZK) proof/argument for unit vector that allows the prover to convince the verifier that a vector of ciphertexts $(C_0, \ldots, C_{n-1})$ encrypts a unit vector $\mathbf{e}_i^{(n)}$, $i \in [0, n-1]$ with $O(\log n)$

proof size. The proposed SHVZK protocol can also be Fiat-Shamir transformed to a non-interactive zero-knowledge (NIZK) proof in the random oracle model.

- **We provide prototype implementation [IOH19] of the proposed treasury system for running and benchmarking in the real world environment.** In Chapter 3, we detail our implementation written in Scala programming language over Scorex 2.0 framework, using TwinsCoin consensus for keeping the underlying blockchain. Main functionality includes proposal submission, registration of voters, experts, voting committee members and their corresponding deposit lock, randomized selection of the voting committee members among voters, distributed key generation (6-round protocol), ballots casting, joint decryption with recovery in case of faulty committee members (4-round protocol), randomness generation for the next treasury period (3-round protocol), reward payments, deposit paybacks, and penalties for faulty actors. All implemented protocols are fully decentralized and resilient up to 50% of malicious participants. During verification, we successfully launched a testnet comprising 12 full nodes operating tens of treasury periods with different parameters.

- **We extend the constructions of the treasury system to develop a privacy-preserving system for general blockchain decision-making.** We extend the applicability of our initial voting protocol within treasury system to support cryptocurrencies with private stake information. e.g. Zcash. Specifically, in Chapter 4, we develop a novel honest verifier zero-knowledge (HVZK) proof for the multiplicative relation between two vector ciphertexts. This proof enables us to demonstrate that two vector ciphertexts satisfy a multiplicative relationship. Hence, we design and implement a practical system that can be deployed for community-inclusive collaborative general-purpose decision-making with support for delegative voting on blockchains with private stake.

- **Decision-making Consensus Evaluation.** In Chapter 4, for the first time, blockchain-based decision-making, to enhance the degree of participation and

quality of decisions reached through the voting systems e.g. treasury system voting, we provide a consensus evaluation system. The system allows individual participants within the system to assess the degree of closeness or agreement of their choices with general network decisions. Thus, enabling and encouraging participants' discussions and awareness, which in turn increases participation and user satisfaction on the outcomes of decisions reached within the system. The main measures we use for consensus evaluation are distance measure, consensus degree, consensus measure, and proximity measure. Distance measure is a measure of the separation between a party's choice and the group/network decision while consensus degree is an aggregation of the distance measure per project. Similarly, consensus measure is an aggregation of the consensus degree using ordered weighted average (OWA). Finally, proximity measure evaluates each party's voting preferences in relation to the collective decision reached on the network by aggregating the distance measure across all proposals/issues/candidates.

1.4 Thesis Outline

The remainder of this thesis is structured as follows:

Chapter 2 presents related work for this thesis. It summarises pertinent issues around blockchain funding and utilisation. Specifically, it covers blockchain funding and alternatives in relation to sustainability and community-inclusiveness in fund utilisation. To capture state-of-the-art, it further develops the presentations in a publication presented at 'The 18th IEEE International Conference on Data Mining (ICDM 2018) Workshop on Blockchain and Sharing Economy Applications (BlockSEA) in Singapore (November, 2018)'.

Further, Chapter 2 presents background information on terminology, describe notations and cryptographic primitives used in subsequent chapters - Chapter 3 and Chapter 4. Furthermore, it covers discussion on Universally composable (UC) [Can00] model of secure computation for analysing security of cryptographic protocols. The UC-model is a framework under the provable security paradigm in cryptography, and was adopted for

proving the security of the systems developed within this thesis.

Chapter 3 presents a treasury system that addresses identified issues in the exploration of blockchain maintenance and funding management discussed in Chapter 2. Essentially, it discusses a high-level abstraction of fundamental operations and attributes of cryptocurrencies. Additionally, it provides preliminary cryptographic concepts of distributed key generation, and zero-knowledge protocols pertinent to the presentation of the proposed treasury system. Furthermore, the chapter covers entities, participants and details the operations and stages of the developed treasury system for cryptocurrencies. The treasury system supports more robust funding sources, enables participation of stakeholders/community members in decision-making process on fund utilisation. In order to optimise community intelligence ('crowd wisdom'), the system supports delegation of voting resources to experienced or more knowledgeable community members (experts) within the voting system. That is, the voting protocol of the treasury system supports delegative voting otherwise known as liquid democracy. The presentations in Chapter 3 is based on a publication presented at 'The Network and Distributed System Security Symposium (NDSS 2019) held at San Diego, California in February 2019'.

Chapter 4 presents a privacy-preserving general decision-making system for blockchain management. The system can be adapted for general decision-making such as change of blockchain rules, transaction and block sizes, etc. Overall, the system builds upon techniques and concepts from the earlier mentioned treasury system in Chapter 3. Specifically, it presents a privacy-preserving decision-making system that supports cryptocurrency with private stake information using novel multiplicative-relation non-interactive zero-knowledge proofs/arguments (NIZK). Furthermore, to enhance utility of decisions and participation amongst stakeholders, we present a (soft-)consensus analysis of decisions reached on the system using distance measures. This enables users to determine the degree of agreement and proximity of individual choices with overall decisions reached on the system. The work presented in this chapter is based on a research work submitted to the IEEE Transactions on Dependable and Secure Computing journal and a report of further analysis of the treasury system available on Cryptology ePrint Archive [ZOB18].

In Chapter 5, we first present an analysis of the security of the proposed systems. This presentation is done by examining how the developed systems satisfy conventional notions of security in the electronic voting literature. Thereafter, Chapter 5 provides information around implementation, benchmarking, testing, and evaluation of the main algorithms, protocols and systems discussed in Chapter 3 and Chapter 4. Mainly, it covers details about frameworks, computational resources, requirements, and libraries used in the implementations of prototypes of the systems.

Chapter 6 concludes this thesis with a reflection on the presentations of the previous chapters of the thesis, limitations and ethical considerations of some design decisions. Finally, the thesis concludes with research directions for future work.

Chapter 2

Background and Related Work

This chapter provides information on cryptographic primitives, concepts, abstractions, and notations used in subsequent chapters. Mainly, these are building blocks of the cryptographic protocols presented in this thesis. Specifically, it covers public key encryption, elliptic curve cryptography and definitions of security. Furthermore, it discusses ElGamal encryption and its additive homomorphic encryption property used in the voting protocol of the treasury system proposed in Chapter 3 and subsequently the general-decision making system proposed in Chapter 4.

Zero-knowledge proofs, a tool used to ensure parties in a protocol execution behave as outlined in the protocol, and critical for verifiable electronic voting is reviewed next. Moreover, the review focuses on a Shamir-fiat transformation of sigma protocols (zero-knowledge proofs) known as non-interactive zero knowledge proofs that are critical to the presentations in this thesis. Thereafter, we present a brief discussion on universally composable (UC) security [Can00], a new simulation-based paradigm of proving security of cryptographic protocols.

The chapter also provides relevant literature review on related work - electronic voting in Section 2.14 and blockhan voting and governance in Section 2.15 - and concludes with a summary of the presentations covered in Chapter 2.

2.1 Terminology and Notations

The following notations are used throughout this thesis. The security parameter is defined as $\lambda \in \mathbb{N}$. Let $[a, b]$ denote the set $\{i \in \mathbb{N} | a < i < b\}$ and $[\ell]$ denote the set $\{i \in \mathbb{N} | 1 < i < \ell\}$. We denote by PPT *probabilistic polynomial time* Turing machines. By $\mathbf{a}^{(\ell)}$, we denote a length-ℓ vector $(a_1, \ldots, a_\ell)$. We use the notation $s \leftarrow S$ to denote uniform random sampling of s s from a set, or distribution. When A is a randomised process, i.e. PPT algorithm, $y \leftarrow A(x)$ denotes running A on input x with a fresh random coin r. When needed, we denote $y := A(x; r)$ as running algorithm A on input x with r as the explicit random coin.

Let $\mathrm{poly}(\cdot)$ be a polynomially-bounded function and $\mathsf{negl}(\cdot)$ be a negligible function. A function is defined as negligible if for every positive polynomial $p(\cdot)$ and all sufficiently large $\lambda's$, it holds that $\mathsf{negl}(\lambda) < 1/\mathrm{poly}(\lambda)$. An event is said to occur with negligible probability if the probability of the event is $\mathsf{negl}(\lambda)$, and an event occurs with overwhelming probability if its complement occurs with negligible probability, that is, $1 - \mathsf{negl}(\lambda)$.

2.2 Public Key Encryption

Formally, a public key encryption scheme consists of three algorithms defined as follows:

Definition 1. *(Public Key Encryption, PKE). A public key encryption (PKE) scheme* PKE *is a triple of PPT algorithms,* $\mathsf{PKE} = (\mathsf{PKE.KeyGen}, \mathsf{PKE.Encrypt}, \mathsf{PKE.Decrypt})$ *defined as follows:*

- ***Key Generation Algorithm,*** $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$*: on input the security parameter* λ *and outputs a secret key* sk *and a public key* pk*.*
- ***Encryption Algorithm,*** $c = \mathsf{PKE.Encrypt}(\mathsf{pk}, \mathsf{m})$*: on input public key* pk *and a message* $m \in \mathcal{M}$*(message space), outputs a ciphertext* c*.*
- ***Decryption Algorithm,*** $m = \mathsf{PKE.Decrypt}(\mathsf{sk}, \mathsf{c})$*: on input secret key* sk *and a ciphertext* c*, outputs a message* m*.*

Following is a presentation of the relevant correctness and security requirements for the cryptographic constructions in this thesis.

Definition 2. *(Correctness). A PKE scheme* PKE *is said to be correct for all* $\lambda \in \mathbb{N}, m \in \mathcal{M}$, $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$, *it holds that*

$$Pr[\mathsf{PKE.Decrypt}(\mathsf{sk}, \mathsf{PKE.Encrypt}(\mathsf{pk}, \mathsf{m})) = \mathsf{m}] = 1$$

Definition 3. *(Indistinguishable Chosen Plaintext Attack, IND-CPA Security) A public key encryption (PKE) scheme ,* $\mathsf{PKE} = (\mathsf{PKE.KeyGen}, \mathsf{PKE.Encrypt}, \mathsf{PKE.Decrypt})$ *is said to be IND-CPA secure, if for any PPT adversary* $\mathcal{A}$, *the following experiments* $\mathsf{IND\text{-}CPA}_{\mathcal{A},\mathsf{PKE},\mathsf{b}}(\lambda)$ *parameterised by* $b \in \{0,1\}$ *it holds that :*

$$\mathbf{Adv}(\mathcal{A}) \stackrel{\text{def}}{=} |\mathsf{Pr}[\mathsf{IND\text{-}CPA}_{\mathcal{A},\mathsf{PKE},1}(\lambda) = 1] - \mathsf{Pr}[\mathsf{IND\text{-}CPA}_{\mathcal{A},\mathsf{PKE},0}(\lambda) = 1]| \leq \mathsf{negl}(\lambda)$$

$\mathsf{IND\text{-}CPA}_{\mathcal{A},\mathsf{PKE},\mathsf{b}}(\lambda)$:

1. The challenger runs $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$ and provides pk to $\mathcal{A}$.
2. $\mathcal{A}$ outputs a pair of messages (m_0^*, m_1^*).
3. The challenger computes $c_b^* \leftarrow PKE.Encrypt(\mathsf{pk}, m_b^*)$ and provides c_b^* to the adversary $\mathcal{A}$.
4. $\mathcal{A}$ outputs a guess bit b' which is the output of the experiment returned by the challenger.

ElGamal [Elg85] is a good example of public key encryptions scheme that is **IND-CPA** secure and this thesis utilises IND-CPA security of ElGamal encryption from the Decisional Diffie Hellman (DDH) assumption.

2.2.1 Elliptic Curve Over $\mathbb{F}_p$

For the purpose of improved efficiency, the implementation of cryptographic schemes in this thesis are based on elliptic curve groups. Mainly, these are groups consisting of

points on elliptic curves rather than groups based on modular arithmetic. Moreover, in contrast to finite field multiplicative groups $\mathbb{Z}_p^*$ there exists no known sub-exponential time algorithms for solving the discrete-logarithm (Dlog) problem in aptly chosen elliptic curve groups [KL14]. Therefore, for cryptographic constructions based on the DDH or Dlog problem such as our treasury protocol, the elliptic curve implementation is more efficient than that based on prime-order sub-groups (DDH problem is hard and Dlog problem are believed to be hardest). Hence, the choice of elliptic curve groups for our implementation to achieve efficient solutions at a given level of security.

Let $\sigma := (p, a, b, g, q, \zeta)$ be the elliptic curve domain parameters over a finite field $\mathbb{F}_p$, consisting of a prime p specifying the finite field $\mathbb{F}_p$, two elements $a, b \in \mathbb{F}_p$ specifying an elliptic curve $E(\mathbb{F}_p)$ defined by $E : y^2 \equiv x^3 + ax + b \pmod{p}$, a base point $g = (x_g, y_g)$ on $E(\mathbb{F}_p)$, a prime q which is the order of g, and an integer ζ which is the cofactor $\zeta = \#E(\mathbb{F}_p)/q$. We denote by by $\mathbb{G}$, the cyclic group generated by g. It is assumed that the DDH assumption holds over $\mathbb{G}$, that is for all PPT adversary $\mathcal{A}$:

$$\mathsf{Adv}_{\mathbb{G}}^{\mathsf{DDH}}(\mathcal{A}) = \left| \Pr \left[\begin{array}{l} x, y \leftarrow \mathbb{Z}_q; b \leftarrow \{0,1\}; h_0 = g^{xy}; \\ h_1 \leftarrow \mathbb{G} : \mathcal{A}(g, g^x, g^y, h_b) = b \end{array} \right] \right| \leq \frac{1}{2} + \mathsf{negl}(\lambda) \ ,$$

Formally,

Definition 4. *(Decisional Diffie-Hellman, DDH). The DDH problem is said to be hard over the cyclic group $\mathbb{G}$ generated by g if for all PPT adversary $\mathcal{A}$ there is a negligible function* negl *such that*

$$|\mathsf{Pr}[\mathcal{A}(\mathbb{G}, q, g, g^x, g^y, g^z) = 1] - \mathsf{Pr}[\mathcal{A}(\mathbb{G}, q, g, g^x, g^y, g^{xy}) = 1]| \leq \mathsf{negl}(\lambda),$$

where uniform $x, y, z \in \mathbb{Z}_q$ are sampled.

2.2.2 Elliptic Curve Lifted ElGamal

This section contains a brief explanation of a variant of ElGamal encryption scheme known as lifted or exponential ElGamal. This variant of the ElGamal scheme is particularly

useful for the cryptographic voting protocol described in Section 3.5. because of its additive homomorphic property.

Lifted ElGamal encryption scheme is used as the additively homomorphic public key cryptosystem in our protocol construction and consists of the following four PPT algorithms:

- $\mathsf{Gen}(1^\lambda)$: on input security parameter λ, and output $\sigma := (\mathbb{G}, p, a, b, g, q, \zeta)$.
- $\mathsf{EC.KeyGen}(\sigma)$: pick $\mathsf{sk} \leftarrow \mathbb{Z}_q^*$ and set $\mathsf{pk} := h = g^{\mathsf{sk}}$, and output $(\mathsf{pk}, \mathsf{sk})$.
- $\mathsf{EC.Enc}_{\mathsf{pk}}(m; r)$: output $c := (c_1, c_2) = (g^r, g^m h^r)$.
- $\mathsf{EC.Dec}_{\mathsf{sk}}(c)$: output $\mathsf{Dlog}(c_2 \cdot c_1^{-\mathsf{sk}})$, where $\mathsf{Dlog}(x)$ is the discrete logarithm of x. (Note that $\mathsf{Dlog}(\cdot)$ is not efficient, therefore, the message space $\mathcal{M}$ should be a small set, e.g., $\{0,1\}^\eta$ for $\eta \leq 30$, to enable efficient exhaustive search of $\log_g g^m$. Moreover, lookup tables are used in practice to enhance efficiency of the decryption process).

As earlier stated, it is well established in the literature that lifted ElGamal encryption scheme is IND-CPA secure under the DDH assumption [Gol04,KL14,DK15,Mol01,Sma03, PP10]. Its key generation and decryption algorithm can be efficiently distributed and it has additively homomorphic property (cf. Section 2.2.3 below).

2.2.3 Additively Homomorphic Encryption

Let $\mathsf{Gen}(1^\lambda)$ be the group generator algorithm that takes as input the security parameter $\lambda \in \mathbb{N}$, and outputs the group parameters param as earlier explained in Section 2.2.2, which defines a multiplicative cyclic group $\mathbb{G}$ with prime order p, where $|p| = \lambda$. We assume the DDH assumption holds with respect to the group generator Gen. Specifically, The additively homomorphic cryptosystem HE consists of four PPT algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Add},)$ defined as follows:

- $\mathsf{KeyGen}(\mathsf{param})$: pick $\mathsf{sk} \leftarrow \mathbb{Z}_q^*$ and set $\mathsf{pk} := h = g^{sk}$, and output $(\mathsf{pk}, \mathsf{sk})$.
- $\mathsf{Enc}_{\mathsf{pk}}(m; r)$: output $c := (c_1, c_2) = (g^r, g^m h^r)$.

- $\mathsf{Dec}_{\mathsf{sk}}(c)$: output $\mathsf{Dlog}(c_2 \cdot c_1^{-\mathsf{sk}})$, where $\mathsf{Dlog}(x)$ is the discrete logarithm of x.
- $\mathsf{Add}(c_1, \ldots, c_\ell)$: output $c := (\prod_{i=1}^{\ell} c_{i,1}, \prod_{i=1}^{\ell} c_{i,2})$.

Lifted ElGamal encryption is additively homomorphic, that is:

$$\begin{aligned}\mathsf{Enc}_{\mathsf{pk}}(m_1; r_1) \cdot \mathsf{Enc}_{\mathsf{pk}}(m_2; r_2) &= (g^{r_1}, g^m h^{r_1}) \cdot (g^{r_2}, g^m h^{r_2}) \\ &= (g^{r_1+r_2}, g^{m_1+m_2} h^{r_1+r_2}) \\ &= \mathsf{Enc}_{\mathsf{pk}}(m_1 + m_2; r_1 + r_2) \ .\end{aligned}$$

We adopt the well known threshold lifted ElGamal encryption scheme as the candidate threshold additively homomorphic public key cryptosystem for our treasury protocol.

2.2.4 Pedersen Commitment

Specifically, in the unit vector zero-knowledge proof, we use the Pedersen commitment first introduced by Pedersen [Ped91] in 1991. The scheme has homomorphic properties and is based on similar algebraic operations and hardness assumptions as the lifted threshold ElGamal encryption used in our protocol. Moreover, it is perfectly hiding and computationally binding under the discrete logarithm assumption.

The Pedersen commitment scheme consists of the following 4 PPT algorithms. Each of these algorithms implicitly take as input the same group parameters, $\mathsf{param} \leftarrow \mathsf{Gen}(1^\lambda)$.

- $\mathsf{KeyGen}^{\mathsf{C}}(\mathsf{param})$: pick $s \leftarrow \mathbb{Z}_q^*$ and set $\mathsf{ck} := h = g^s$, and output ck.
- $\mathsf{Com}_{\mathsf{ck}}(m; r)$: output $c := g^m h^r$ and $d := (m, r)$.
- $\mathsf{Open}(c, d)$: output $d := (m, r)$.
- $\mathsf{Verify}_{\mathsf{ck}}(c, d)$: return valid if and only if $c = g^m h^r$.

Pedersen commitment exhibits additively homomorphic property, i.e.

$$\begin{aligned}\mathsf{Com}_{\mathsf{ck}}(m_1; r_1) \cdot \mathsf{Com}_{\mathsf{ck}}(m_2; r_2) &= (g^m h^{r_1}) \cdot (g^m h^{r_2}) \\ &= (g^{m_1+m_2} h^{r_1+r_2}) \\ &= \mathsf{Com}_{\mathsf{ck}}(m_1 + m_2; r_1 + r_2) \ .\end{aligned}$$

2.3 Schwartz-Zippel lemma

Here, we briefly discuss a variation of the Schwartz-Zippel lemma [Sch80] that is used in proving the soundness of our zero-knowledge protocols.

Lemma 1 (Schwartz-Zippel)**.** *Let f be a non-zero multivariate polynomial of degree d over $\mathbb{Z}_p$, then the probability of $f(x_1, \ldots, x_n) = 0$ evaluated with random $x_1, \ldots, x_n \leftarrow \mathbb{Z}_p$ is at most $\frac{d}{p}$.*

Therefore, given two multi-variate polynomials f_1, f_2, if

$$f_1(x_1, \ldots, x_n) - f_2(x_1, \ldots, x_n) = 0$$

for random $x_1, \ldots, x_n \leftarrow \mathbb{Z}_p$, then we can assume that $f_1 = f_2$. This is because, if $f_1 \neq f_2$, the probability that the above equation holds is bounded by $\frac{\max(d_1,d_2)}{p}$, which is negligible in λ.

2.3.1 Zero-knowledge Proofs/Arguments

Zero-knowledge (zk) proof is a fundamental cryptographic building block and have wide application in general multiparty computation protocols [GMW87] such as secure verifiable voting protocols [Adi06]. Generally, cryptographic zero-knowledge proofs lets a party (the prover) prove/convince another party (the verifier) about an assertion or truth of a statement without revealing any extra information. Mainly, the security of zero-knowledge proofs are characterised by three main properties, namely, *zero-knowledge, soundness*, and *completeness*.

Let $\mathcal{L}$ be an NP language and $\mathcal{R}_{\mathcal{L}}$ be its corresponding polynomial time decidable binary relation, i.e. $\mathcal{L} := \{x \mid \exists w : (x, w) \in \mathcal{R}_{\mathcal{L}}\}$. We say a statement $x \in \mathcal{L}$ if there is a witness w such that $(x, w) \in \mathcal{R}_{\mathcal{L}}$. Thus, statement x and witness w and only x is given to the verifier V. Let the prover P and the verifier V be two PPT interactive algorithms. Denote $\tau \leftarrow \langle P(x, w), V(x)\rangle$ as the public transcript produced by P and V. After the protocol, V accepts the proof if and only if $\phi(x, \tau) = 1$, where ϕ is a public predicate function.

Definition 5. *We say (P, V) is a perfectly complete proof/argument for an NP relation $\mathcal{R}_{\mathcal{L}}$ if for all non-uniform PPT interactive adversaries $\mathcal{A}$ it satisfies*

- *Perfect completeness:*

$$\Pr\left[\begin{array}{l}(x, w) \leftarrow \mathcal{A}; \tau \leftarrow \langle P(x, w), V(x)\rangle : \\ (x, w) \notin \mathcal{R}_{\mathcal{L}} \vee \phi(x, \tau) = 1\end{array}\right] = 1$$

- *(Computational) soundness:*

$$\Pr\left[\begin{array}{l}x \leftarrow \mathcal{A}; \tau \leftarrow \langle \mathcal{A}, V(x)\rangle : \\ x \notin \mathcal{L} \wedge \phi(x, \tau) = 1\end{array}\right] = \mathsf{negl}(\lambda)$$

Let $V(x; r)$ denote the verifier V is executed on input x with random coin r. A proof/argument (P, V) is called *public coin* if the verifier V picks his challenges randomly and independently of the messages sent by the prover P.

Definition 6. *We say a public coin proof/argument (P, V) is a perfect special honest verifier zero-knowledge (SHVZK) for an NP relation $\mathcal{R}_{\mathcal{L}}$ if there exists a PPT simulator* Sim *such that*

$$\Pr\left[\begin{array}{l}(x, w, r) \leftarrow \mathcal{A}; \\ \tau \leftarrow \langle P(x, w), V(x; r)\rangle : \\ (x, w) \in \mathcal{R}_{\mathcal{L}} \wedge \\ \wedge\, \mathcal{A}(\tau) = 1\end{array}\right] \approx \Pr\left[\begin{array}{l}(x, w, r) \leftarrow \mathcal{A}; \\ \tau \leftarrow \mathsf{Sim}(x; r) : \\ (x, w) \in \mathcal{R}_{\mathcal{L}} \wedge \\ \wedge\, \mathcal{A}(\tau) = 1\end{array}\right]$$

Sigma protocols ($\sum$) are three-move proof protocols [DJN10] with a challenge-generating verifier such as Schnorr [Sch91] and Chaum and Pedersen [CP93] used to efficiently prove knowledge of Dlog (discrete logarithm) and exponents in a Diffie-Hellman triple, respectively. At a high level, in the first move, the prover sends (commitment to) a random value to the verifier, then the verifier sends a randomly generated value to the prover. Finally, to convince the verifier, the prover sends a final value which is a combination of the original value (in the first move), the challenge (in the second move), and the witness to the verifier.

A public coin SHVZK proofs/arguments can be transformed to a non-interactive one (in the random oracle model [BR93]) by using Fiat-Shamir heuristic [FS86] where a cryptographic hash function is used to compute the challenge instead of having an online verifier in the protocol interaction. We provide more details in Section 3.7 of the zero-knowledge proofs used in the treasury system protocol.

2.3.2 Universal Composability

We model our proposed treasury system security under the standard *Universally Composable* (UC) security framework [Can00] proposed by Canetti in 2000. UC provides a framework for proving security of cryptographic protocols, and is, perhaps, the most famous proof paradigm under simulation-based security proofs for cryptographic protocols (alternative to game-based or property based definitions of security). A key concept of the framework is to enable stand-alone security definitions of protocols that still holds under concurrent general composition.

The main technique of UC is to create an ideal functionality $\mathcal{F}$ that describes the desired security goals of a protocol [BU13]. Following the specification of an ideal functionality, we then develop a protocol that is as secure as the ideal functionality, in a process known as 'UC emulation'. The designed protocol is secure if it 'UC-emulates' the ideal functionality. In other words, a protocol is secure for a given task, if no adversary attacking an execution of the protocol can gain more from the attack than attacking an ideal process involving a trusted third party (that gets input from the parties and hands

output back to them) running the ideal functionality for the protocol.

This approach to analysing security of protocols was first proposed in [GV87]. Essentially, the approach states that a secure protocol for a computational task should 'emulate' an ideal process for the task. In turn, an ideal process for a computational task involves a 'trusted third party' that receives input from all parties and locally executes the tasks, computes the output and hands said output back to the parties. In the UC framework, an environment $\mathcal{Z}$ is responsible for generating and handling inputs to parties and receiving outputs whilst communicating arbitrarily with the adversary.

Furthermore, the UC framework enables secure composition, i.e. given a secure protocol ρ with ideal functionality $\mathcal{F}$, protocol ρ is still a secure protocol if ideal functionality $\mathcal{F}$ is replaced by another protocol π that UC-emulates $\mathcal{F}$. Thus, enabling composition of protocols. Moreover, the UC framework guarantees that a protocol proven secure in a stand-alone model will remain secure when executed in an environment with other concurrently run arbitrary protocols. This is otherwise known as achieving concurrent general composition [Lin09].

Variants of UC framework include simplifed-UC (SUC) [CCL15], a simpler and restricted framework with built-in authenticated channels, fixed parties, and less complications than the generalized UC framework. Other variants are UC with joint state (JUC) [CR03] where protocol instances have some joint state and randomness, UC with global setup (GUC) [CDPW07] where protocols are analysed in a setting with parties having access to trusted global information with specific properties. Others are synchronous-UC [KMTZ13] that allows UC to support protocols achieving properties such as input completeness, guaranteed termination, etc. through the introduction of (ideal functionalities for) loosely synchronised clocks and bounded latency (in communication channels/networks). Synchronous-UC provided a novel formal approach for achieving synchrony in the UC framework.

Under the framework, all entities (protocols, functionalities, environment) including adversarial entities are represented as interactive Turing machines (ITMs), each of which represents the program to be run by participants. Note that, we distinguish between

ITMs (which represent static objects, or programs) and *instances of ITMs* (*ITI*s). ITIs represent interacting processes in a running system initiated by an environment that communicates with the adversary, hands input to and gets outputs from the protocols. Moreover, an ITI is an ITM along with an identifier that distinguishes it from other ITIs in the same system. The identifier consists of two parts: A session-identifier (SID) that identifies which protocol instance the ITI belongs to, and a party identifier (PID) that distinguishes among the parties in a protocol instance. Typically, the PID is also used to associate ITIs with "parties" that represent some administrative domains or physical computers. The pair (SID,PID) is a unique identifier of the ITI in the system. The model of computation consists of a number of ITIs that can write on each other's tapes in ways specified in the UC model. We assume that all ITMs are PPT.

Specifically, the security of our voting protocol is assessed in the UC framework with static corruption in the random oracle (RO) model. The security is based on the indistinguishability between real/hybrid world executions and ideal world executions, i.e. for any possible PPT real/hybrid world adversary $\mathcal{A}$, we construct a PPT simulator $\mathcal{S}$ (ideal world adversary) that can present an indistinguishable view to the environment $\mathcal{Z}$ operating the protocol.

2.4 Background

Blockchain research has witnessed significant advances since the pioneer blockchain Bitcoin was invented a decade ago. From first generation blockchains that are largely passive distributed ledgers that facilitate token exchanges between potentially distrusting parties [Nak08, Tak18, DD14], to second generation blockchains or programmable state machines [Woo14] that support contracts expressed in logical computer codes, i.e. smart contracts. Advances in blockchain research and practice have mainly focused on improved security and privacy [DD14, Max13, KKKZ19, BCG$^+$14, KMS$^+$16], scalability [ASB$^+$18, KJG$^+$18, ZMR18, KZ19, GKZ19, BCD$^+$14], sustainability and interoperability [LXS$^+$19]. Recent blockchains are rife with some or a combination of these features. They now offer better scalability, privacy, security and support better transaction throughputs than first

generation systems.

However, beyond these improvements, 'inclusive' blockchain development and management is another main goal amongst modern cryptocurrencies [Hor18, Har16, Mer16, But, DFL16, Th16, KNS+16]. Intuitively, this enables stakeholders of a cryptocurrency (beyond developers, founders, investors) to have active roles in the management of development funds for blockchain, and general decision-making that affects overall health of these distributed platforms. The focus of this thesis aligns with the latter goal by designing and developing provably-secure protocols that support inclusive management, development and general-decision making in cryptocurrencies.

Real-world cryptocurrencies require continuous maintenance through developmental projects and steady funding for survival and continuous growth. Development of early blockchains and cryptocurrencies e.g., Ethereum [Woo14], Dash [DD14], ZeroCash [BCG+14], were mainly funded by 'founders', who are developers, investors, researchers, etc. Consequently, cryptocurrency development and growth is largely influenced, driven and determined by founders. There abound ample evidences of failed cryptocurrencies and disputes due to undemocratic control by 'founders/owners' of these supposed distributed platforms [DuP17, DFL16, Hac19].

However, blockchain-based treasury systems have been identified as an alternative funding and decision making mechanism for distributed ledgers [KKN+,KN+,KKN+17,ZOB18, Th16, Hor18]. A treasury system is a self-sustenance mechanism for the advancement of blockchain systems [KKN+17]. Generally, a blockchain treasury system covers sourcing of funds, funding management and decision-making for fund utilisation to support cryptocurrency maintenance and development. Moreover, it is a decentralised, community-inclusive collaborative decision-making system for cryptocurrencies and blockchains. Mainly, it comprises regular (mostly decentralised) funding source, community-inclusiveness and (incentivised) participation, collaborative, secure, periodic, and ideally an enforcement (punishment) mechanism to ensure compliance with system-wide decisions. Section 2.5 provides a high-level description of treasury systems for cryptocurrencies.

Cryptocurrency projects such as core development, light-client support, mining

pool software, decentralised exchanges, research etc. requires funding to execute. 'He who pays the piper dictates the tune', therefore, Section 2.6 examines funding sources for cryptocurrency maintenance and development and their effect within treasury and cryptocurrencies.

Treasury-like functions of key selected cryptocurrencies were further explored for insights towards design decisions in our proposed blockchain treasury system, developed and presented in Chapter 3. Analysis of details of operations, features, strengths and weaknesses of these blockchain funding management systems is presented in Sections 2.8, 2.9, 2.10 and 2.11. Particularly, the analysis examines how these pioneer blockchain platforms address pertinent problems such as community-inclusiveness, participant spread, voting rules, privacy, security, fund sourcing, fund utilisation, etc. and their effects in developing secure treasury systems for sustainable blockchain technologies. Moreover, Section 2.13 discusses funding sources in relation to fund utilisation and decision-making process, and provides a taxonomy of these (treasury-like) system functions based on existing cryptocurrencies and beyond.

Note that due to the alluring nature of cryptocurrencies [MSA19] as a store of value and attendant community interest, a considerable volume of work in this area originate from non-academic, industrial [WZX+18] or grey literature such as whitepapers, e.g., the Bitcoin whitepaper [Nak08], official websites, third-party service providers e.g., Coinmarketcap, official and unofficial chat forums e.g., Reddit, Medium, mailing lists, Wiki pages, GitHub, etc. Moreover, practice is sometimes ahead of theory in blockchain research [BMC+15, WZX+18]. Accordingly, the presentation in this chapter also draws directly from these, largely unofficial, sources.

Subsequently, we provide a

Finally, Section 2.16 provides concluding remarks by highlighting key findings and insights, and provides a summary of research questions that are addressed in subsequent chapters of this thesis.

2.5 Treasury System Overview

Generally, cryptocurrencies require skills spanning various fields such as cryptography, economics, law, finance, software development etc. for tasks and projects such as marketing, secure protocol design and proof, code for system core, business registration, advertising and publicity, etc. Due to decentralisation on blockchains and absence of trusted-third parties, ideally, extra care is required to prevent introduction of centralisation through these 'external' sources. Therefore, a treasury system seeks to address these sources of centralisation from the operations of the cryptocurrency. For instance, by allowing community stakeholders contribute to the system.

A treasury system is a 'self-sustenance' mechanism for decentralised distributed ledgers. Mainly, it encompasses blockchain funding, funding distribution, utilisation, as well as decision-making, [KKN+17, KKN+, ZOB18, Car18a, Hor18, KNS+16] for the development, maintenance, advancement and long-term sustainability of underlying cryptocurrency. A cryptocurrency treasury system represents a decentralised means of achieving and ensuring sustenance and improvement of its parent-system through guaranteed regular funding for supportive developmental projects and democratised decision-making.

Moreover, a treasury system covers how project proposal submissions are made, community discussions, community-inclusive collaborative decision-making for reaching decisions on proposals, and reliable robust funding for the treasury. Desirably, treasury system parameters and features should be compatible with, and reflect considerations for real-world scenarios in which the blockchain community and proposers are expected to function. For example, it can mimic salary payment in the real-world by adopting monthly cycles for all of its activities. Below, we highlight key requirements for treasury systems [KNS+16, ZB18, Dasb] to help diminish centralisation in the operations of cryptocurrencies.

- *Treasury funding mechanism.* Although blockchains can generate money for holders through appreciation in exchange value, they cost money to develop and main-

tain [Bit17, Bita, Bas, Ara, ICO]. Hence, a secure and reliably sustainable source of funding is essential for a treasury system to successfully fund projects that support cryptocurrency development. Potential funding sources include block rewards taxation [HBHW17, DD14], voluntary donations [Bit17], and minting [ZOB18] which is particularly useful when deductions from block rewards become small. Block rewards may reduce as a result of decrease in block rewards known as halving in Bitcoin [NBF+16, Nak08], or as a result of money supply limit being reached and only transaction fees making up block rewards.

- *Community inclusive and stakeholder participation.* Holders of stake and cryptocurrency community members should be allowed and encouraged to participate in the treasury system to ensure sufficient buy-in and minimise concentrating blockchain operations in the hands of a minority few. Game-theoretically efficient *incentive structures* can further drive increased stakeholder participation in treasury process [KN+], and also help ensure that incentives do not lead to hyper-inflation which can diminish cryptocurrency value.

- *Collaborative decision-making mechanism* e.g., a voting scheme. Treasury systems should integrate mechanisms that facilitate community-inclusive collaborative decision-making in order to maximise the collective-intelligence of the cryptocurrency community in particular and larger blockchain ecosystem [KN+, ZOB19]. In addition the collective-decision making mechanisms should be flexible [Tez] and reflect the peculiarities of the blockchain rather than general voting rules, to enhance utility and participation in the system. For instance, a higher threshold can be set for decisions on projects that exceeds a defined funding target, or for proposals with high blockchain forking risks [KKN+17] rather than simple majority voting rule for all proposals.

- *Blockchain and stakeholders' security and privacy.* Participation in treasury activity should not compromise the security and privacy of participants and the cryptocurrency. For instance, the system should be secure against Sybil attacks [GKL15],

rushing attacks [KMS+15], nothing-at-stake attacks [BNPW18] and stakeholders' votes should be weighted relative to their stake in the system.

Next, we examine the main sources of development funding (treasury funding mechanism) for cryptocurrencies in relation to risks, benefits and sustainability for treasury systems and blockchains in general. The remaining above-listed requirements are covered in Chapter 3, which details the proposed cryptocurrency treasury system with community-inclusive decision-making mechanism that exploits collaborative intelligence through delegative voting (cryptocurrency treasury system). We remark that, ideally, project milestone compliance monitoring and/or refund mechanism in case of non-compliance are desiderata for cryptocurrency treasury systems.

2.6 Funding Sources

Funding is essential for the development and maintenance of any real-world cryptocurrency. For example, human resources (salaries and compensation), computing resources (hardware and software), marketing, legal fees, etc. are some necessary expenses accrued in the development and maintenance process of real-world blockchain. Main sources of development funding on blockchain systems include donations (charity), taxations, venture capitalists (investor funding), patron organisations and more recently Initial Coin Offerings (ICOs) [GMV20, HS18]. Today, cryptocurrencies largely rely on one or a combination of these sources for development funding and maintenance. The next Sections 2.6.1, 2.6.2, 2.6.3, 2.6.4, 2.6.5, present an examination of each of these funding sources.

2.6.1 Initial Coin Offerings (ICOs)

ICO is a means of crowd-funding done by issuing some cryptocoins to investors, organisations, or individuals, in exchange for other cryptocurrencies, usually Bitcoin or Ethereum, or a fiat currency e.g, US Dollars. In recent years, ICOs have become a substantial means of funding cryptocurrencies and other blockchain-based startups and

businesses [CHSW19, GMV20]. Typically, most ICOs are deployed as smart contracts that receive funds and distribute equivalent amount of the new or underlying cryptocurrency to the investors at a later date [Bita]. Technologies such as the ERC-20 [VB15] which facilitate easy development of new tokens used for ICOs have enabled increase in the number of ICOs in the cryptocurrency community. Usually, investors buy these tokens in expectation that their value may rise if project/company offering the ICOs are successful in the future [GMV20, MSA19]. ICOs represent a good way to raise reasonable amount of money to support cryptocurrency development within a short period of time. For instance, projects such as Basic Attention Token [Bas] and Aragon [Ara] were able to respectively raise 35 million USD and 25 million USD in 30 seconds and 15 minutes [Kea17, CD17, ICO].

Due to the successes recorded by early ICOs, there has been a rise in the number of ICOs for cryptocurrencies and other blockchain-based technologies [ICO]. According to Cointelegraph [VB15], approximately 4 billion USD (fiat currency equivalent) was raised in 2017 alone. While ICOs are good, there are uncertainties associated with the legality of ICOs. For example, are ICO tokens regarded as payment token, utility tokens or financial security [HT18, OEC19]? This classification is necessary for global regulators in relation to protection of investments in ICOs [GMV20]. Unfortunately, ICOs can also be abused by fraudsters who abuse the minting process that creates new coins used in the ICOs [CHSW19] and have been used in Ponzi schemes and financial scams [WA17] because of the low entry requirements (mainly technical costs) to organise an ICO as evidenced by the plethora of over 250,000 ERC-20 tokens [Eth20] on the Ethereum blockchain. ICOs are susceptible to concentration of token ownership in the hands of a minority, speculation, and scams. Moreover, recently there have been an apparent decline in the rate of successful ICOs, perhaps due to an increase in the number of speculative projects issuing ICOs.

Evidently, ICOs are valuable for initial development or starting out a project. This is attributable to the relatively high amount of funds that can be raised in a short span of time, especially as sources of low cost finance [OEC19]. However, uncertainties

surrounding the amount of funds that can be raised at any given time [GMV20] make them unsuitable for solely funding of cryptocurrency treasury system. Treasury systems require stable, reliable and largely predictable funding sources [ZOB19,KKN+17,KNS+16] to support cryptocurrency development. ICOs are incompatible with the methodical planning and reliable funding that is expected of decentralised cryptocurrency treasury systems. Therefore, an obvious drawback is that ICOs are not suitable as sustainable source of funding for long-term planning of cryptocurrency maintenance and development.

2.6.2 Donations

Donations is one of the oldest and most common means of cryptocurrency development funding. Often developers/inventors/founders of a cryptocurrency rely on donations from charity e.g., from the public, stakeholders, or other third parties, for cryptocurrency development funding e.g., Bitcoin Core Sponsorship [Bit17]. These donations can be in form of cash money (cash) or through offering of services such as software development, or marketing skills (kind). For instance, Bitcoin through its foundation- The Bitcoin Foundation (whose goal is to increase global acceptance of Bitcoin) - have developed via reliance on donations and membership registration fees. For the Monero (XMR) [Mon] cryptocurrency, developers or proposers (planning to execute cryptocurrency development projects) willing to contribute to the growth of Monero, seek donations from other community members by discussing details of their proposals (including the amount of funds needed) on the 'getmonero forum' [Get].

Moreover, BitcoinCore, an open source endeavour responsible for the maintenance of the Bitcoin client [Bitb], also runs a sponsorship program through which top companies and industry players can contribute to the development of the cryptocurrency. Although, donations represent an alternative funding mechanism to ICOs, they are not without their own pitfalls. There have been instances where development activities have been threatened because of low availability of funds. For instance, in 2015, a board member published a note on the Bitcoin Foundation forum where he stated that the Foundation was seriously struggling with funds.

Although, donations are useful for cryptocurrency development funding, solely, they are not suitable for long-term methodical planning and sustainability of cryptocurrencies due to associated fund uncertainties [KNS+16]. Meticulous planning for blockchain development activities is impractical under the donations only model due to uncertainties about the amount of funds that will be available at any period of time. Furthermore, donations could introduce centralisation in the system. For instance, popular donors (not necessarily malicious) may take advantage of the model to influence decisions in the system to their favour. In contrast, a malicious adversary or donor may try to influence the system in ways that are detrimental to the sustainability and growth of the system. For instance, this may be done to cripple the system due to the adversary holding a large stake in an altcoin or competitor cryptocurrency.

2.6.3 Taxations

Taxation is an alternative development funding source found on existing cryptocurrencies such as Dash [DD14], ZCash [Zca], etc. By design, a fraction of block rewards, payment for finding new block, is taken and contributed to a decentralised pool, or the cryptocurrency company/founders, or some other bodies providing services to the cryptocurrency. For instance, 10% of miners' rewards on the Dash blockchain is contributed to a central pool. Developers and other community members can request funds from this pool for proposals that will advance the Dash blockchain. On ZCash, the 'founders reward' is 10% of the 21 million ZEC that will be mined on the network. This represents tax that will be distributed among founders, investors, employees, and advisors for services offered to support the blockchain.

Taxation, perhaps, is the most sustainable of all cryptocurrency development funding sources because it is a blockchain sustenance mechanism derived from activities on the blockchain itself. Furthermore, it lends itself to methodical planning which are required by disintermediation protocols such as cryptocurrencies and does not have the uncertainties associated with ICOs [GMV20]. Therefore, taxation which is obtainable from miners' rewards and potentially transaction fees represent a good treasury fund source. However,

additional funding schemes may be required because of decreasing miners' reward, also known as halving in Bitcoin [NBF+16, Nak08], which is a common feature of most cryptocurrencies. Accordingly, to supplement the shortage from diminishing miners' reward, a hybrid treasury source, such as one that combines taxation with donations or minting represents a more robust and sustainable source of blockchain funding.

2.6.4 Minting

Minting is defined as the process of creating new cryptoassets [CHSW19] or cryptocoins to mainly support cryptocurrency development [ZOB19]. On the Bitcoin blockchain, coinbase transactions [NBF+16] are used to create new Bitcoins. Basically, this is the means by which miners that propose new blocks get rewarded. The miner proposing the new block receives some new cryptocoins for being the first to correctly solve a computational puzzle (proof-of-work) or for being the leader of a committee responsible for proposing a new block [KRDO17] (proof-of-stake). That is, nodes are rewarded for providing critical services to the network. Similarly, minting would create new coins for solely supporting cryptocurrency development by paying or rewarding community members who propose and carry out projects that will advance the growth of the blockchain. In other words, newly minted coins are used solely to fund projects (that support cryptocurrency development) by awarding the minted coins to community members with successful proposals.

Although minting provides stable funding for cryptocurrency development, it creates overall inflation in the value of underlying cryptocurrency [CHSW19] and susceptible to abuse by proposers who are only interested in taking funds from the system because of the 'seemingly unlimited' availability of funds supply. Therefore, there is need for careful adoption, implementation and thorough economic and game-theoretic analysis to determine the optimum amount of coins that should be minted at any given time towards funding projects in the system. This analysis is also necessary to determine the overall inflation impact and implications for long-term growth and value of the cryptocurrency. Hence, to protect against adverse inflation, it is required to achieve an equilibrium between

supply and demand, and Cohney et al., rightly posits that stakeholders participation is necessary to attain optimal minting values [CHSW19].

2.6.5 Patron Organisations

Apart from ICOs, taxations, donations, and minting, patron organisations also provide services to support blockchain development. Patron organisations provide dedicated support for cryptocurrency development through the provision of support services. For instance, Bitpay contributes to Bitcoin development by funding wages of some Bitcoin Core developers. Furthermore, industrial organisations can also contribute human resources and services to the growth of a cryptocurrency. For instance, in addition to providing funding to Bitcoin Core, Blockstream [1] also contribute human resources to the Bitcoin cryptocurrency e.g. developers contributing to Bitcoin Core software. Patron organisations tend to wield ample influence in decision-making on the development direction for cryptocurrencies. Issues generally arise when the decisions reached do not reflect the inputs of the broader cryptocurrency community. Usually, these decisions tend to occur in a centralised manner, and centralised sustenance mechanisms that do not proportionately represent the interests of all stakeholders are not appropriate for blockchains.

In summary, it is clear that a clever combination of the above funding sources considered with the peculiarities of given cryptocurrencies would provide adequate funding source for secure, decentralised cryptocurrency treasury for supporting system development. Each of the above-listed blockchain development funding sources, individually does not provide a secure, methodical and robust funding source for decentralised blockchain-based cryptocurrencies' treasury system.

2.7 Related Work

Here, we shed more light on the operations of key real-world cryptocurrencies (Dash, Zencash, Decred, and Zcash) in relation to fund sourcing and utilisation through com-

[1] https://blockstream.com/

munity participation. This exploration helps to identify gaps that are addressed in our proposed treasury system presented in Chapter 3. Besides, e-voting, blockchain voting and governance are topics closely related to the work presented in this thesis. Therefore, we provide a review of these topics in Section 2.14 and Section 2.15 respectively. The review provides a window into the state-of-the-art in literature around key topics related to the presentations in this thesis. Specifically, we explore the literature on developments of e-voting systems and place a strong emphasis on end-to-end verifiability, before concluding with discussions of blockchain-based voting and blockchain governance.

2.8 Dash Governance System

The Dash governance system (DGS) [Dasb] also referred to as Dash governance by blockchain (DGBB) is a pioneering treasury implementation for cryptocurrency development funding in any real-world cryptocurrency [KNS+16]. The DGS allows regular users on the Dash network to participate in the development process of the Dash cryptocurrency by allowing them submit project proposals that support the cryptocurrency to the network. A subset of 'special' nodes known as Masternodes then vote to decide what proposals receive funding. Masternodes are a subset of dedicated servers with a 1000 Dash that provide special services such as instant and private pay, to the Dash network [DASa]. Every voting cycle (approximately one month), winning proposals are voted for and funded from the accrued resources in the blockchain treasury. 10% of all block rewards within each monthly voting period is contributed towards a pool (*superblock*) from which proposals are then funded. Essentially, the DGS uses taxation as the means of development funding and consists of three major stages: proposal stage, voting stage, and payment stage.

Next, we provide descriptions of the DGS by discussing the proposal and decision-making processes.

2.8.1 Proposal Submission and Decision-making

Projects willing to support the Dash cryptocurrency request funds from the network by submitting proposals through a special transaction known as GOVERNANCE_OBJECT_PROPOSAL [KNS+16]. This transaction burns 5 Dash to prevent Denial-of-Service attacks on the network. Proposals include name, URL to detailed proposal, date (start and end date for funding request), payment address (if proposal is successful after voting), and the amount of funds requested.

Proposal transactions are stored 'locally' by nodes on the Dash network after verifying their validity because proposals are not stored on the Dash blockchain. Following submission, proposers are expected to publicise their submissions to Masternodes who vote on proposals within the Dash Blockchain. There exists third-party tools and websites [2] to support campaigning, tracking and voting for proposals.

For decision-making on DGS, masternodes can vote, without the need for representatives or delegates in a direct democracy process [Hel06, LIJ84, Tor05, Vat00]. The voting ballot options on proposals are 'YES', 'NO', or 'ABSTAIN', respectively implying support 'for', 'against' or 'neutrality'. Voters on the DGS vote directly on proposals in the system using the Dash wallet or other third-party tools or websites. Voting is done via a special transaction that identifies the masternode as well as the hash of the proposal that is being voted on.

Ballots are also stored locally by the nodes on the network [KNS+16]. Similar to the proposal submission process, every node on the network checks to confirm the validity of the votes before adding to their internal storage. Note that, like proposals, votes are also not stored on the blockchain.

In the payment stage, in order to be considered for funding at the end of voting, accrued votes for any proposal must satisfy the fuzzy threshold voting rule [KNS+16]. The rule states that the number of net positive votes for any winning proposal must be greater than 10% of all masternodes on the network. Specifically, for Dash governance system, the equation below captures the requirement necessary for any proposal to be

[2] www.dashcentral.org, http://dashmasternode.org/masternode-tools/, https://proposal.dash.org/, https://dashnexus.org/, https://www.dashninja.pl/governance.html

successful:

$$V_{\mathsf{votes}} = (V_{\mathsf{yes}} - V_{\mathsf{no}}) \geq 0.1 * |\text{masternodes}|$$

where V_{yes} is the number of yes votes, V_{yes} represents the number of votes in support (yes votes), V_{no} represents the number of votes against (no votes) and |masternodes| is the number of masternodes on the Dash network. That is, only proposals with V_{votes} are eligible for funding.

Finally, all eligible proposals are then ranked in descending order according to the amount of net votes received and are funded accordingly until the available budget is exhausted. Note that, once the available monthly budget is expended, other eligible proposals are discarded. Therefore, only top-ranked proposals that fit within the available budget are funded in an automated process that requires synchronisation of the proposal rankings among masternodes and communication on the network. As earlier mentioned, the total budget for any voting period is gotten from taxation of miners' block rewards, and is calculated as follows:

$$\mathsf{Budget} = (\text{Block reward} * 0.1) * 16616 \text{ blocks}$$

2.8.2 Discussion

For the first time in cryptocurrencies, DGS provided a mechanism for supporting cryptocurrency development from itself and allowed nodes on the network participate in the process of determining who gets funding to support the system. However, the system can be improved by addressing some of the issues highlighted below.

The reliance on off-chain proposal and voting data (local storage) in the processes of DGS presents a potential issue that makes the system susceptible to validation attacks, thereby, potentially leading to forks on the blockchain. Also, participation in the DGS voting process is not incentivised, despite resources (time and computing resources) expended by masternodes. Hence, there is not a high turnout in masternode participation in the DGS [KNS+16] given that only masternodes (excluding other stakeholders) can vote in the system. For example, a successful proposal titled *DASH-CORE-GROUP-LEGAL-*

EXPENSES [3] received 752 Yes, 59 No, and 15 Abstain votes, which is approximately 18% turnout with an estimation of 4500 masternodes.

Moreover, voting on the DGS is not private, as a result, voters can be subjected to coercion from a powerful adversary. Furthermore, given the fact that miners' reward decreases by about 7.14% annually on the Dash blockchain, supplementary funding sources for the DGS is necessary. This is because reliance on taxation alone may not be sufficient for future budgeting considering projected increase in community size and corresponding increase in number of proposals.

Meanwhile, proxy (delegation) voting or liquid democracy has emerged as an alternative model for community-wide decision-making that can reduce participation cost and time especially for apathetic voters. Liquid democracy is a hybrid of direct and representative democracy that allows voters to directly vote on issues or select a representative that they delegate their voting power/right to [For02, KKH+15, CG17, ZZ17a, KR20]. Therefore, liquid democracy can enable community collective (rather than masternode only or selective) participation in the DGS voting process. An obvious advantage is that it enables the community to take advantage of members with expertise within a community i.e. optimise the wisdom of the crowd. Also, it enables increased participation, belonging and acceptance from community members because of its inclusiveness. This is in addition to the scaling advantage it provides to the system. That is, when the system grows and there are enormous amounts of proposals to vote for, it will be practically impossible for all members of the community to properly scrutinise the merits of all submitted proposals. Therefore, reliance on experts through vote delegation, can help more community members make confident voting decisions.

2.9 ZenCash Treasury

The ZenCash treasury (now transitioned to Horizen), by design, adopts a flexible multi-stakeholder governance model [VVL17]. The core idea is to remove centralisation which entrusts enormous power to a minority group. Participation is voluntary and

[3] https://www.dashcentral.org/p/dash-core-group-legal-expenses

decision-making power cuts across all categories of stakeholders proportional to their resources(stake).

Initially, the ZenCash system has a Core Team (inclusive of founders of Zen) and a Decentralised Autonomous Organization(DAO) (consisting of industry leaders) that controls 3.5% of block mining reward and 5% of rewards respectively. The plan is to evolve, develop and adopt a hybrid voting mechanism that enables all stakeholders to influence decisions and resource allocations on the blockchain. This evolution would result in a system of DAOs, with competing DAOs responsible for working on different problems. Collectively, the DAOs will be responsible for activities (building, maintaining, improving software, legal, marketing, and advertising) that will ensure the long-term sustainability of Zen.

The staff strength of each DAO is between $3 - 5$ members and could potentially be increased to any number. Community members / stakeholders are allowed to participate in the development of Zen via project proposals which are obviously funded by the DAOs through the 5% block mining reward allocation they receive. Proposals are only funded subject to successful voting and only one DAO existed at launch.

2.9.1 Proposal Submission and Decision-making

Proposal submission is done at no cost and available to any interested party [VVL17]. There is a proposal submission deadline of two weeks before commencement of voting and starting period of one day after voting. The proposed governance mechanism has a bimonthly (6 times in a year) voting period. Voting is done with the use of tokens, with a cap of 1400 tokens available at any voting period. The tokens are distributed as follows:

- 360 tokens are available for sale to stakeholders
- 240 tokens are reserved for ZenCash project developers (based on some metrics e.g., number of commits or pull requests, etc.)
- 60 tokens are given to cryptocurrency exchanges that support ZenCash
- Another 60 tokens are provided for mining pool owners

- 360 tokens are for Secure Nodes
- 120 tokens are distributed equally among DAO officers and
- Similarly, the final 240 tokens are shared among members of the core team

The system proposes a majority and super-majority voting (winning) rule of votes greater than 720 and 1080 respectively out of the available 1400 tokens/votes. However, the core team has a veto power over all proposals with a unanimous vote by all members of the core team.

2.9.2 Discussion

Unlike Dash, voting is open to a wider range of stakeholders (including parties that purchase tokens) instead of only a special subset of nodes, although with the limitation of a fixed number of tokens. Community participation is further limited because stakeholders can only hold 25% of all available voting tokens. In comparison, the DAOs and Core Team also have the same amount of voting power towards the decision-making process. Moreover, the proposal submission process is susceptible to DoS attacks because it does not cost tokens/money to make submissions [VVL17].

The ability of veto power granted to the core team comprising only 3 individuals can lead to an excessive concentration of power in the hands of a minority. Potentially, this power can be abused against decisions reached by the larger voting community i.e. 'tyranny of the powerful/founders'. The utility of the system to adapt to a larger audience and optimise community knowledge is hindered by its voting structure. That is, the system does not support delegative voting and only few members could participate due to the limited amount of voting tokens.

Finally, there are plans to address some of the identified issues within ZenCash [VVL17] and evolve governance structure into a more encompassing and effective system.

2.10 Decred Treasury

Initially, the Decred cryptocurrency [Deca] also deployed a governance system like DGS(in the Dash network). However, unlike the DGS, an organization known as Decred Holdings Group, LLC (DHG) is responsible for management of 'treasury funds'. Specifically, this fund is known as development subsidies and is obtained by taxation of 10% miners' rewards [KNS+16]. Amongst other duties, the DHG is responsible for publishing periodic financial statements, calling for proposals, and submitting budget proposals. Within the DHG there are various 'Councils', e.g., Admission Council, Attrition Council, that oversee appointing community members to the Decred Assembly. The Decred Assembly is responsible for voting on proposals and decide on projects to be funded in a process described as Proof-of-Assembly (PoA).

However, in 2018, Decred released Politeia (Pi), a platform that supports its governance. The platform allows community members to submit, discuss, vote and track proposals. This allows stakeholders to participate in fund utilisation decisions and let anyone to contribute to the development of Decred. Areas of contribution include development, marketing, research, or community. To ensure sustainable funding for community contributions to development of Decred, Politeia also relies on contribution of 10% of all block rewards(subsidies) to a treasury pool(taxation). The key goal is to prevent a powerful minority from exercising disproportionate influence in the development of the cryptocurrency projects [Decb]. Contributions are funded as Politeia proposals after they have been created and recognised by other Contractors (members whose work have been funded in the past). As a Decentralised Autonomous Organisation (DAO), stakeholders vote on budgets and policies [Deca].

2.10.1 Proposal Submission and Decision-making

Parties willing to carry out projects that request funds from the treasury submit their proposals in a transaction that burns 0.1 Decred. Submitted proposals are checked for validity and invalid proposals are removed by Politeia administrators. Politeia uses transparent censorship to prevent malicious invalidation of proposals. However, this presents

a potential bottle-neck in the system when the number of proposals grow exponentially. Voting on any project is triggered by an administrator following authorization by the proposer and spans 2016 blocks (about 1 week). For approval, at the end of voting a proposal must get:

- 60% 'Yes' votes and
- Voter turnout of at least 20%

Once a proposal is approved, the owner can begin execution of the projects and can claim funds on submission of deliverables and manual verification by Decred Holdings Group (DHG).

To participate in the voting, stakeholders must lock some Decred in return for a voting-ticket. After obtaining tickets, it takes 256 blocks (approximately 20 hours) to become eligible for participation in the voting process. Ticket price is set every 144 blocks (about 12 hours) to maintain a target of about 40,960 total tickets (ticket pool). Random eligible tickets are selected to vote 'Yes' or 'No' for proposals. If a ticket holder misses the voting window (e.g., due to being offline), their tickets become invalid and their funds returned after voting ends. To avoid missing voting, individual users can register with Voting Service Providers [Decc] who charge a fee (up to 5%) for helping individuals relay their voting choices to the network (through a 1-of-2 multi-signature script). While VSPs do not have access to ticket funds, they receive voting rewards are share to parties accordingly. Voters receive some rewards in addition to their locked funds after voting. Voting tickets selection follows a Poisson distribution with an average of 28 days before being selected to vote.

2.10.2 Discussion

The Decred approach to funding utilisation enables the DHG to implement useful checks such as payment of contractors in arrears and verification of milestones of proposals and projects. Although, data from the Politeia platform are committed to public git repos and anchored to public the Decred blockchain using its time-stamping application,

dcrtime, the Decred treasury system relies on trusted parties within its operations, e.g., proposal validity check, voting commencement, off-chain voting etc. Thus, potential problems arising from reliance on trusted party for approval include conflict of interest, coercion of members, risk of fund treasury fund theft and overall, introduction of a single point of failure (to a desirably decentralised system).

Heavy reliance on human intervention at every point in the project funding process, makes the system prone to inefficiencies and subtle attacks and abuse and complicates the scaling process when there is a large growth in the number of community of users participating in the process. Furthermore, it is quite difficult to adequately capture or model the various ways in which users would interact with the system. Thus, making a thorough security analysis of the system non-trivial.

Furthermore, the Voting Service Providers provide a form of delegated voting, however, it is reliant on the VSPs to correctly relay user choices to the network (without guarantee they would relay user's preferences). Within the Decred treasury, the main purpose of delegation is to sustain user/system liveliness rather than optimisation of skills, experience and intelligence of 'expert' voters in the system. Currently, VSPs control about 37% of all tickets in the system (with 12.95%, the largest held by a single VSP), thereby leading to potential centralisation risks and overwhelming influence in voting outcomes.

The problem is further exacerbated given that voting ballots are not private, thereby, potentially leading to voting bandwagon effect [SB96] (where users vote for projects that are apparently winning regardless of their merit). Other attacks on the voting process include collusion and coercion due to the open ballot system.

The irregularity of the voting windows (beginning and ending time) and voter ticket selection/activation makes it complex for individual voters efficiently participate and track the whole process. Hence, encouraging the participation in VSP's, which in turn leads to further centralisation of the voting process through VSPs. Moreover, the DHG relies on off-chain resources thereby introducing external threats to the system.

2.11 Zcash Governance

Zcash is a privacy-centric proof of work mining-based cryptocurrency that utilises zk-snark as its fundamental building block. Zcash was founded by the Zero Electric Coin Company [Zca] and is the organisation responsible for developing the protocols of the Zcash cryptocurrency. The company serves the interests of founders, investors and developers of the Zcash coin. 20% of all block mining rewards are paid to the company under a scheme termed Founder's rewards.

Through the Zcash Foundation, an independent body from the Zero Electric Coin Company, community members can propose projects to support the development of Zcash. Particularly, the Zcash foundation through its GitHub Grant Proposal Page [Fou] requests community members to submit proposals that fall within the foundation's defined scope, as well as support the overall mission and vision of the foundation.

The Zcash Foundation Grant runs a quarterly funding cycle. The Zcash foundation board oversee the process and decides the amount of funds approved towards community-sponsored projects. During the first month, calls for proposals are released and deadlines for initial submissions are approximately one month from the date of call for proposals. Following this, discussions and critiquing of received submissions are then expected to take place. Final submissions reflecting feedbacks from the discussions are then expected to be submitted approximately twenty-one days from the initial submissions. Thereafter, the Zcash Foundation Grant Review Committee review the proposals and provide funding decisions (which also requires the approval of the Zcash Foundation Board) in approximately one month.

Payment is made in a single lump sum in the ZEC equivalent of the fiat currency requested in an approved proposal. As a means of ensuring transparency and future evaluations, proposers of funded projects are required to provide progress reports six months after receiving funding. In Q4-2017, only 5 members voted in the round of budget approval that approved 300 ZEC (approximately $\$80,000$) as the amount available for community-driven projects. Nonetheless, at the board's discretion, it has power to fund projects in excess of the approved budgeted amount.

However, there are plans for changes to the funding process since the acceptance of the Zcsah improvement proposal (ZIP) 1014 [4] which directs the funding of Zcash protocol development and privacy research. Specifically, a Major Grants Review Committee (MGRC) comprising 5 members who are elected (using a Helios poll) by a 'community advisory panel' [5] assembled by the Zcash Foundation, will now be responsible for distributing the 8% grant constituent of the taxation from miners' reward on the Zcash blockchain.

2.11.1 Proposal Submission and Decision-making

Community members with proposals wishing to be considered for funding need first to submit an informal detailed description of their proposals (on GitHub). Following this, based on comments, critiques, suggestions and feedback from other community members, proposal owners submit a revised formal version of their initial proposals (also on GitHub). Zcash Foundation Grant Review Committee (5 members) appointed by Zcash foundation then considers all submitted proposals and decide on the winning proposals (that receive funding) based on comments and the proposals (or by making additional consultations where necessary).

Proposal submissions are expected to contain information about the following [Fou]: Motivation and overview, technical approach, team background and qualification, plan, security considerations, schedule, budget and justification. The details provided under each of the listed sections are the basis upon which comments, critiques, and suggestions are raised before a final proposal submission is made.

Final formal proposal submissions are to be made via attachments to the original (informal) submissions on the 'Zcash Foundation Grants: Call for Proposals'. In summary, the submission process involves users submitting their proposals, and community members providing comments, and suggestions for improvements. Proposal files are issues (on the Zcash foundation dedicated GitHub repository) which contains details about the proposals.

[4] https://zips.z.cash/zip-1014
[5] https://bit.ly/2HxleEN

2.11.2 Funded Proposals

In October 2016, the first funded project was a contest (Zcash Open Source Miner Challenge), operated by LeastAuthority.com (a company that supports Zcash Electric Coin Company), to support projects that would encourage wide and open source mining of the ZEC in order to further strengthen the Zcash cryptocurrency. $\$30,000$ was paid in rewards to the top 5 winning projects selected by a panel of 3 judges.

The first set of of funding made by the Zcash Foundation (from revenues obtained from the Founder's rewards) was the award of 33 ZEC each to three recipients under the 'Test Transaction Awards'. This was an award made to three vibrant members of the Zcash community who have consistently developed tools and solutions that support Zcash. Following the award, announcement about the launch of the Zcash Foundation Grant program was made to encourage other community members to participate in the development of the cryptocurrency whilst taking advantage of available funding.

Although, the process raised awareness about the Zcash Foundation Grant scheme, the community was not a part of the decision making process in selecting the deserving members. Similarly, the community also played no part in the process of selecting the winning proposals of the Zcash Open Miner Challenge.

2.11.3 Discussion

Clearly, the Zcash approach to treasury would not scale for a cryptocurrency with a large number of proposals to consider. When the Zcash system grows and there are hundreds of proposal submissions, it is practically infeasible for each member of the Foundation Grant Review Committee to scrupulously review each proposal to determine its merit for funding approval. Moreover, relying on a select number of reviewers appointed by a board (that grants final approval and also determines the amount of funds available for funding proposals) is not representative of a decentralised, open and inclusive development system for cryptocurrencies. Clearly, a better and alternative approach would be to accommodate a larger number of community members (or possibly all willing members) in the decision-making process.

Having, a 5 member committee examine all proposals on a decentralised blockchain system is clearly problematic, particularly considering that the members of the committee can also submit proposals of their own. For example, in a funding round, one of the members was unavoidably absent, thereby limiting the team strength to 4. Some proposals were reviewed by only 3 reviewers due to conflict of interest by one of the review team members who also submitted a proposal. Note that although the Zcash Calls for Proposals states that Review committee members must exclude themselves from the discussion of their own proposals, there are still subtle ways in which a committee member can still influence their proposals towards getting funded. For instance, a committee member who has submitted a proposal and has knowledge of the approved budget for proposals, can deliberately down-vote or negatively review other projects under consideration, such that total budget request of all approved proposals is less than the available budget. Thereby, increasing the chances for the acceptance of the member's proposal.

Furthermore, while the feedback process of the initial submission is desirable and helps the overall quality of proposals submitted to the system, only a limited amount of members from the large Zcash or cryptocurrency community participate in this process. Definitely, a system that encourages participation among all community members, such as delegative democracy (a collaborative decision-making mechanism) represents an improvement over the approach deployed in the Zcash Foundation Grant process. Another significant drawback of the funding mechanism of the Zcash Foundation Grant system is the non-use of the Zcash blockchain at any point in the decision-making process. Funding decisions on a blockchain-based cryptocurrency should leverage the resources of the blockchain to improve the decision-making process. For, instance, secure time-stamping of final proposals, commitment(hashing) of proposal submissions, voting in the decision making process, tamper-proof 'locking' of voting and funding decisions, etc.

Moreover, at any point in future, it is not particularly clear how much money would be available for funding proposals. In other words, the amount of community-driven projects or proposals is solely dependent on the approved budget by the foundation board. Although, the Zcash Foundation Board is empowered to make discretionary approvals

above a budget for any particular quarter, uncertainty about available budget may inhibit the amount of development proposals that the community can contribute to the overall development of the cryptocurrency. Hence, an alternative solution, such as 'decentralised treasury pool' represents a better solution for the system.

In conclusion, the existing system will evolve with time and some of the issues raised will be addressed towards achieving an improved system. For instance, the establishment of 'Working Groups' for monitoring volunteers/proposers is a proposed addition to the system. Furthermore, information on the Zcash Foundation Grant GitHub page also corroborates the suggestion that there are plans to evolve the system into an 'open-community review process'.

2.12 Ethereum Foundation Grants

So far, we have discussed Dash, Decred treasury and other cryptocurrency treasury systems. Nonetheless, there are cryptocurrencies that do not adopt treasury system. Therefore, we explore the Ethereum blockchain where much discussion and analysis revolve around governance system rather than treasury system. Note that a governance system is not the same as a treasury system, although, both systems support long-term sustainability of blockchain technologies. Basically, treasury system handles and ensures regular funding and decision-making on fund usage whereas a governance system is mainly concerned with general blockchain and protocol rules such as soft-forks and hard-forks. A governance system guides how and when changes are made to a blockchain's core.

Similar to the Bitcoin Foundation, the Ethereum Foundation is a not-for-profit organisation that supports Ethereum through research, development and education [Woo14]. Like the Bitcoin Foundation, the Ethereum foundation also relies on donations as its primary source of funding for Ethereum development and research. Basically, the foundation uses donations to support 'general' Ethereum development activities. However, donors can specify specific projects for which their donations can be used for (provided the foundation supports such projects).

Although, the Ethereum blockchain does not have a treasury system, it supports

development projects through grants to support research that would improve blockchains. For example, the programs referred to as subsidy programs were established to support projects on sharding and layer-two protocols that would improve blockchain scalability. Teams, researchers, developers, who do relevant work in this area can apply for support through a process described as 'flexible to accommodate various needs of different applicants'. The Ethereum core leadership is responsible for deciding what proposals are funded, with funding amounts of $\$50,000$ up to $1 million. Clearly, this encourages open and community-wide participation, however, the core leadership wields much power in the decision-making process.

Critics of decentralised governance argue that adaptation of blockchain protocols to changes is slow on systems with decentralised governance due to the minimum requirements for approval being high. They argue that decentralisation constitutes the major point of arguments against existing off-chain or traditional blockchain governance model. Critics further argue that a side-effect of decentralised on-chain governance is the ability of a rich minority (with very substantial amount of stake) to have over-bearing influence on decision-making via a voting process.

Other highlighted criticisms or suggested factors that hinder on-chain voting governance are low participation (or voter turnout) and unequal wealth distribution. Within Ethereum, for instance, for Ethereum Improvement Proposal (EIP) 186 titled 'Reduce ETH issuance before proof-of-stake' [6], voting was conducted through Carbonvote [7]. With approximately 2.7 million ETH in participation, it had less than 20% (quorum) voter turnout [8]. Proposal 17 [9] with about 10% of the quorum has the highest voter turnout of all received proposals on the DAO given about 23,606 unique Ethereum addresses holding tokens (DTH - DAO Token Holders) [10] at the time of voting. However, low-voter turn-out is not peculiar to blockchain voting. According to FairVote, established democracies such as the United Kingdom and United States also experience low voter

[6] https://github.com/ethereum/EIPs/issues/186
[7] http://carbonvote.com/
[8] https://hackernoon.com/notes-on-blockchain-governance-ob65o3pod
[9] https://daostats.github.io/proposals.html
[10] http://themerkle.com/the-dao-undergoes-low-voting-turnout/

turnout. For instance, only about 55.67% of eligible voters participated in the 2016 US presidential elections [KKH+17]. Similarly, the 2015 UK elections recorded about 66.6% voter turnout [11].

However, the problem of low voter incentive and low voter turnout can be addressed through clever incentives engineering, by rewarding participants who take part in elections. Furthermore, novel voting mechanisms such as liquid democracy mitigate some of the well-known raised issues that affect traditional voting schemes. For instance, these issues are addressed by delegative/liquid democracy where users effectively assign their voting powers to other (better qualified) members of the voting community. Members who receive delegation usually possess more subject-matter expertise and knowledge, and are renowned within the ecosystem.

Having explored various fund sourcing and utilisation, i.e. treasury, on selected cryptocurrencies, following is a presentation of a taxonomy of the different existing systems into treasury models. Although the proposed ontology for treasury models is motivated by current solutions found in existing cryptocurrencies, the classification does extend to cover potential new treasury systems that could be adopted by other cryptocurrencies.

2.13 Taxonomy of Treasury Models

A key motivation for the classification of treasury systems is to identify their properties, common and distinguishing features, strengths and weaknesses towards facilitating further understanding and analysis of the existing diverse groups in real-world cryptocurrencies. Basically, it is to support the adoption of these systems based on their properties, applicability and peculiarities of different blockchain technologies. A major overarching attribute used in the classification of treasury systems is the source of development funding for projects, as well as the input and contribution of community members of the cryptocurrency ecosystem (especially, people who hold stake in the system) in the process of fund utilisation.

[11] https://www.parliament.uk/about/how/elections-and-voting/general/

Specifically, we categorise three main models of treasury systems:

- Open-System Treasury: This class consists of systems where the source of blockchain development funding are external to the system, e.g, Donations, Patron organisations, industry, founders' resources, etc. Additionally, participation in treasury activities such as decision-making, proposal submission, proposal review, and project execution, etc., are open to other members of the cryptocurrency, rather than the cryptocurrency founders, developers and company only.

- Closed-System Treasury: Under this category, all fund sources are local to the system, i.e., within/from the cryptocurrency. e.g, minting, taxation, and donations from stakeholders of the cryptocurrency. Typically, decision-making and cryptocurrency development processes are organised in a centralised manner (e.g., by some organisation) void of input from the general cryptocurrency ecosystem.

- Hybrid-System Treasury: This category of treasury systems represents a combination of the features of the closed and open-treasury systems. Here, cryptocurrency development funding comes from sources within the cryptocurrency and from sources external to the system.

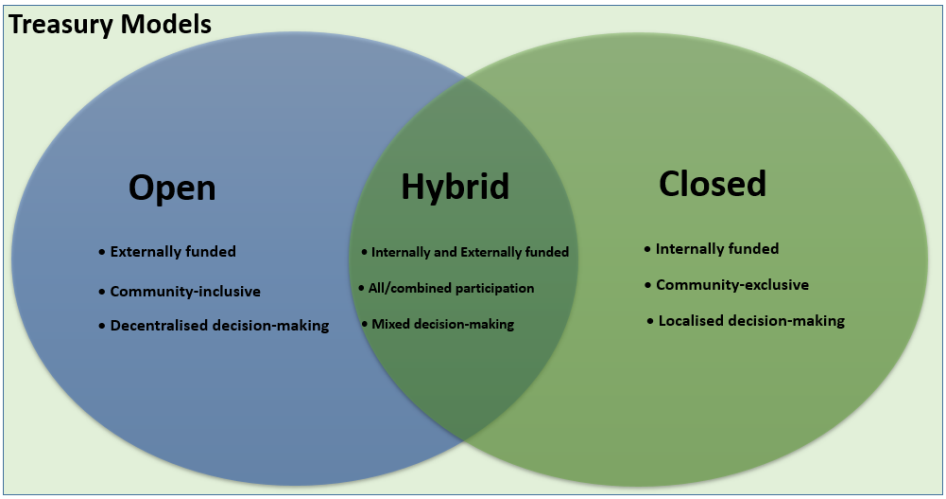

Figure 2.1: Models of Cryptocurrency Treasury Systems.

Deriving from the above-listed treasury models, other concrete sub-classes are highlighted

below.

- **Taxation-Community-Controlled (TCC)**

  Taxation is the primary source of project funding on TCC treasury systems. Primarily, funds are obtained by taxing miners' block rewards or block transaction fees or a combination of both. The Dash governance system is a classic example of a treasury system within the TCC sub-class because it relies on taxations of 10% of block rewards for funding treasury (or fund pool). Decred is another cryptocurrency that relies on 10% miners. reward taxation for self-funding of development projects.

  Within this sub-class of treasury system, project funding decisions are reached through the use of community-inclusive mechanisms such as voting. In other words, funding decisions are made in a decentralised manner (which prevents excessive centralisation of powers in the hands of a few members). A variety of voting systems and rules can be adopted for decision-making such as liquid democracy, preferential voting, plurality voting, approval voting, etc.

- **Taxation-Organisation-Controlled (TOC)**

  TOC like TCC relies on taxation for development/treasury funding. However, unlike TCC, decisions on how to use funds within the treasury are made in a centralised manner. Typically, an organisation or group of individuals (usually company founders or investors) are responsible for making funding decisions, with little or no input from other members of the cryptocurrency ecosystem or cryptocurrency stakeholders.

- **Donation-Community-Controlled (DCC)**

  Under DCC, the major source of cryptocurrency development funding comes from donations. This donation can be obtained from community members, patron or charitable organisations with interest in the development of a particular blockchain. For instance, Bitcoin Core receives donations from individuals and corporate organisations, for carrying out activities that support the continued existence and growth of the Bitcoin cryptocurrency.

Similar to TCC, usage of funds within this treasury sub-class is decentralised. Funding decisions for proposals are made through the use of community-inclusive mechanisms, thereby supporting collaborative decision making.

- **Donation-Organisation-Controlled (DOC)**

  Within this group of treasury system, funding available for cryptocurrency development are sourced from donations. That is, like DCC, DOC treasury systems are also funded from donations received from charities and corporate bodies or community members. For example, proposers of projects that support the Monero cryptocurrency source for funding from donations made by other community members.

  Unlike DCC, DOC treasury decisions are made similarly to that of TOC, i.e., centralised funding decision-making.

- **Hybrid-Community-Controlled (HCC)**

  As the name suggests, HCC relies on open and closed (hybrid) funding sources. Therefore, HCC treasury systems are funded from a combination of sources such as taxation, minting and donation.

  Similar to other community controlled treasury systems, HCC treasury systems use decentralised decision-making mechanisms to reach funding decisions and community members of the cryptocurrency are major participants in the treasury process.

- **Hybrid-Organisation-Controlled (HOC)**

  The source of funding for this sub-class of treasury system is similar to the HCC sub-class explained above. However, a major difference between the HCC sub-class of treasury system and the HOC sub-class is the funding decision making process. Within HCC treasury systems, funding decisions are made in a centralised manner. In other words, the community members or holders of stake in the cryptocurrency wield (little or) no power in the decision making process on how funds are used.

Typically, an organisation responsible for the blockchain or cryptocurrency makes all the decision regarding usage of funds within the HOC sub-class of treasury systems.

Table 2.1: Table summarising the various treasury sub-classes based on funding and decision-making rights.

Feature / Category	Funding		Decision-making Power	
	Open	Closed	Community	Organisation
Taxation-Community-Controlled (TCC)		✓	✓	
Taxation-Organisation-Controlled (TOC)		✓		✓
Donation-Community-Controlled (DCC)	✓		✓	
Donation-Organisation-Controlled (DOC)	✓			✓
Hybrid-Community-Controlled (HCC)	✓	✓	✓	
Hybrid-Organisation-Controlled (HOC)	✓	✓		✓

2.14 Electronic voting systems

Election systems can be deployed wherever there is a need to vote e.g., student union election, general board meetings, parliamentary elections, etc. and the security, auditability, and verifiability requirements largely depend on the system uses-case. Conceivably, the requirements for a national election are more stringent than the election of executives of a local student organisation at a university. The introduction of punched-card ballots in the $1960's$ and direct-recording electronic voting machines in the $1970's$ [Jon03] signalled the revolution towards the use of technology in elections and eventually Internet voting (e-voting) as we know it today.

Chaum in [Cha82] mentioned how his techniques for anonymous communication can be adapted for private e-voting systems. Chaum's solution for pseudonymous sending and receipt of massages enabled the cryptographic design of voting schemes. In addition to Chaum in [Cha82, Cha88], Yao in [Yao82], Benaloh in [Ben87] pioneered the earliest cryptographic voting schemes. Burmester and Magkos in [BM03] provided a classification of e-voting systems in the literature into four main models while Damgard et. al [DGS03] provided a classification with three main models by combining the techniques of two

models (mix-nets and blind signatures) into one. Following is a brief description of the models.

1) *The mix-net model*: Voting schemes under this model use several connected servers called mixes [Cha81, SK95, OA00] to provide unlinkable permuted encrypted ballots as outputs, from encrypted input ballots. Essentially, mix servers produce ciphertexts from input ciphertexts in a random manner with knowledge of the permutation (needed to match input ciphertexts to output ciphertexts) [JCJ05] known to the servers only [BG02]. Multiple rounds of shuffling and decryption by independent mixers is critical to ensure unlinkability between inputs and outputs of the mix-nets and therefore require vast amounts of computation [Fur04]. Examples of e-voting systems under this model are [PIK93,SK95,HS00,JJR02,Nef01,BG02,FMS10]. Despite their well-established universal verifiability property, mix-nets rely on trusted servers and mixing operations can become prohibitively time consuming [SP06].

2) *The blind signatures model*: Voting schemes under this model use blind signatures, first proposed by Chaum in [Cha82] for secure untraceable payments (and anonymous communication), to provide unlinkability between voters and ballots through a validator that blindly signs encrypted ballots. Subsequently, voters unblind the signatures and send encrypted ballots and the 'new' signatures to a voting authority, who in turn posts same to a bulletin board. Anonymous channels e.g. mix-nets, are used to transmit unblinded ballots to the tallying authority to protect voter privacy. Ensuring that the authority learns nothing about the ballot [DGS03]. A bulletin board is basically a public broadcast channel with memory [CGS97, SP06] for communication among voters and voting authorities. This model heavily relies on a honest validator (although can be distributed using threshold cryptography), and substantial multiple rounds of communication between voters, voting authority and the bulletin board. Additionally, ballots are decrypted by the voting authority and tally is posted to the bulletin board. Malicious validators can compromise the security and correctness of voting results by impersonating valid absentee voters [CC97]. Examples of e-voting schemes other this model include [FOO92, OMA+99, GPZ17, Oka97, PHM95].

3) *The Benaloh's model*: Unlike the blind signature model, voting schemes under the Benaloh model utilise $n > 1$ voting authorities. Under this model voters share their ballots (votes) among n authorities using a homomorphic secret sharing scheme [Cha04], and post encryption of the shares under the public key of the recipient voting authority to a bulletin board. Essentially, ballots are 'verifiably secret shared' among the authority servers responsible for tally computation. A t-out-n homomorphic secret sharing is used to achieve robustness such that any number of authorities less than t have no information about a user's ballot. At the end of voting, the set of voting authorities can jointly compute the tally without leaking any additional information. However, there is high communication overhead under for voting schemes under this model. This is as a result of each voter sharing/casting their votes across n multiple channels for the n voting authorities [BM03]. Consequently, this introduces significant practical limitations due to its complexity and high communication overhead because it requires every voter to communicate with every authority in an election. Examples of schemes under this model include [Ben87, CF85, Sch99].

4) *The homomorphic encryption model*: Under this model, additive homomorphic encryption algorithms such as lifted ElGamal [Elg85] or Paillier [Pai99] are used to achieve universal verifiability and individual ballot privacy. Voting schemes under this model focus on hiding the ballot content (through encryption) rather than voter identity or anonymous voting channels, thereby enabling easy verifiability [LGT$^+$03]. Further details on lifted ElGamal encryption is provided in Subsection 2.2.2. Typically, users encrypt their ballots (choice) with the public key of a voting authority (or authorities - where threshold cryptography [Des94] is used), and post the ciphertexts to a bulletin board. A user's choice is represented by a number [CGS97,DGS03], usually, a 1 signifying a yes/support or 0 or -1 signifying a no/against vote. In addition to the encrypted ballots, users also post a proof of validity of the corresponding content (plaintext) from which the encrypted ballot was produced. In other words, voters must prove (in zero-knowledge) that the ciphertexts contain valid ballots (a $0, 1$ or -1), in addition to proving their eligibility to vote. The election authorities obtain the tally by combining (multiplying) individual ciphertexts

(encrypted ballots) without needing to decrypt individual encrypted ballots. Examples of schemes under this model include [BT94, CGS97, BFP$^+$01, DJ01, Adi08, KMNV16]. *Particularly, this thesis is related to the homomorphic encryption model which is discussed in more details in the rest of this section.*

Damgard et al. [DGS03] summarise informal security properties of e-voting systems. Namely, privacy requires that only final results are released or leaked; robustness ensures that all valid ballots are considered in tally computation; *universal verifiability* enables anyone to verify correctness of results; receipt freeness prevents voters from proving how they voted and address vote buying; coercion prevents an adversary from forcing voters to vote a in a certain way. Mitrou et al. in [MGKQ03, MGK02] identified that secrecy is a requirement for voter freedom and expression. Secrecy, essentially the inability to link a ballot to a voter, is a distinctive tenet and requirement of a free political decision-making system and modern democracy. Lambrinoudakis et al. in [LGT$^+$03] provided a list of requirements for voting systems: accuracy, democracy (eligibility), privacy, robustness, verifiability, uncoercibility, fairness, and verifiable participation (in some scenarios). Elections are critical to democratic systems because they allow the general public express their views [KAK18]. Therefore, the process should be transparent and voters and auditors alike should be convinced of the outcomes. Hence, the requirement for end-to-end verifiability (E2E-V).

Verifiability covers mechanisms for auditing an election to ensure that it was correctly conducted [LGT$^+$03]. It can be provided in two different forms: a) *Universal or public verifiability* first introduced by Sako and Kilian in [SK95] through the assumption of physically untappable communication channels between mixing centers and voters, for extending the mix-net based construction of Park et al. [PIK93] (which only provides individual verifiability. Universal verifiability covers the requirement that anyone can verify the correctness of the outcome of an election. b) *Individual verifiability* covers the requirement that every voter can verify their ballot has been correctly accounted for in the election outcome.

In the literature, only few cryptographic e-voting protocols are provably secure [BM03].

Multiple research [KSRW04, FHF07, WWH+10, SFD+14] reveal security vulnerabilities that compromise integrity of some early e-voting systems. Burmester [BM03] highlights the need for secure and efficient cryptographic techniques for e-voting protocols. Consequently, newer voting schemes with E2E-V property have been developed [Cha04, Nef04, AM16]. However, there are no UC-secure decentralised decision-making system.

Given the identified vulnerabilities from analysis of earlier e-voting systems [KSRW04, FHF07], E2E-V e-voting schemes such as [Cha04, Adi08, Ben06] were proposed to address the concerns. A comprehensive review of end-to-end-verifiable voting protocols is provided in [AM16]. The work in [JMP13] provides a high-level survey of verifiability and privacy in e-voting systems. Sampigethaya et al. in [SP06] proposed a framework and a set of metrics for comparing properties of e-voting systems. Cortier et al. in [CGK+16] provides a survey of several notions of verifiability in the literature.

2.14.1 End-to-end Verifiability

The importance of end-to-end verifiability (E2E-V) in electronic voting has been widely established in electronic voting literature. Primarily, voters in any election deserve to be convinced of the correctness of the outcome. Notably, election schemes with E2E-V allow individual voters or interested parties (universal) check crucial parts of election tally without the requirement of trusting any entity in the system [BRR+15]. That is, E2E-V election systems provide mechanisms for voters and external auditors to verify election outcomes and that all valid votes were counted even if devices/servers/election authorities are malicious [CGK+16]. Benaloh et al. [BRR+15] suggests E2E-V technologies as pre-requisites to mitigate inherent risks associated with e-voting systems and a strong security requirement for adoption of e-voting systems [PKRV10, KZZ15b]. The two main techniques/checks for E2E-V are *votes are cast as intended* and *votes are tallied as cast*. *Recorded as cast* is an additional technique that is also found in literature [KZZ15b]. Together, these techniques ensure voters can get evidence that their encrypted ballots reflect their choices by checking for their ballots on a 'public bulletin board' [BRR+15],

and verifying that the election tally includes all valid ballots on the public list. Moreover, 'homomorphic encryption' for tallying without first decrypting individual ballots, and 'verifiable mixing' for permuting and unlinking ballots with voters' identity before ballot decryption, are the two approaches for achieving the *tallied as recorded* requirement.

To achieve E2E-V some voting schemes allow users to generate multiple ballots and submit only one for the tally process. The remainder of the ballots are used as challenge ballots (Benaloh challenge) to verify that the ballots encryptions match the voter's intention. The Helios [Adi08] voting system uses the Benaloh challenge or 'audit or cast' procedure [KZZ15a] to achieve verifiability, while Wombat voting system [BFL+12] and StarVote [BBE+13] combines the approach with paper trail audit mechanism for verification. The scheme in [BPW12] is a Helios variant in the single-pass voting model of [BCP+16] that provides formal guarantees of ballot privacy.

[PKRV10] proposed the first formal *performance requirements-based* definition for E2E-V elections instead of *design requirements-based* definition. The requirements are largely similar to those previously explained in [BRR+15]. Popoveniuc et al. identified the following six checks as conditions any E2E-V election must satisfy. 1) Presented ballots are well-formed 2) Cast ballots are well-formed 3) Ballots are recorded as cast 4) Ballots are tallied as recorded 5) Consistency in ballots recorded and ballots tallied and 6) Every ballot is subjected to Check 3.

Prêt à Voter [CRS05] uses a paper-based system to achieve E2E-V with two-part ballot and permuted order of candidates (that differs from one ballot to another), and uses a mix-net for the tally process involving the permutation seeds. Xia et al. in [XSHT08] analysed a Pallier encryption variant of [CRS05] and demonstrate that voter's choices can be leaked in certain scenarios. PunchScan [PH10] is also an E2E-V e-voting scheme that also combines a paper-based (two-stacked sheet of papers) mechanism.

Furthermore, Scratch&Vote [AR06] is also a paper-based scheme and uses a ballot style similar to [CRS05] but adds 2D bar code representing encryption of all possible user voter's choice. It uses a homomorphic scheme for tallying of encrypted ballots.

Scantegrity II [CCC+08] extends Scantegrity [CEC+08] originally introduced by

Chaum et al. which requires physically locating ballots for dispute resolution. It uses an optical scan ballot with symbols in invisible ink. The ballots reveal a code (receipt which is then publicly published) when marked that users can look up to verify the inclusion of their ballot. Tally is produced using a two-mix scheme similar to PunchScan [PKRV10]. For its security, Scantegrity II requires trusted computing platform for election officials and secure chain-of-custody for paper ballots prior to voting. Helios is a general Benaloh-challenege based E2E-V scheme that uses homomorphic tallying of encrypted ballots.

The E2E-V schemes of Remotegrity/Scantegrity [ZCC+13] rely on randomness beacon to verify correctness of result. Therefore, a biased beacon can compromise the integrity of the system [KZZ15b]. Cortier et al. in [CS11] successfully compromised ballot privacy in Helios using ballot replay attacks, and also proposed a countermeasure for the attack.

Gardner et al. in [GGR09] explored coercion in end-to-end voting protocols and proposed a provably coercion-resistant scheme which extends Benaloh's scheme [Ben06] by asking voters an additional question. However, the proposed scheme relies on a private channel, and that for every election, at least one of the candidates is honest.

Neff [Nef04] introduced a coercion-resistant universally verifiable voting protocol that utilises a receipt to enable any voter verify that their ballots are cast and recorded as intended in the election tally. Chaum [Cha04] introduced a voter-verifiable e-voting scheme (even in the presence of compromised election computers/records) that also utilises physical receipts. Both schemes rely on the temporal availability or destruction of some information. Karlof et al. [KSW05] identified possible methods to leak vote privacy in the schemes of [Nef04, Cha04] which are the first e-voting schemes to allow voter's verify their ballots without computational devices.

Kelsey et al. in [KRMC10] introduced methods for carrying out coercion attacks on paper-based E2E-V e-voting systems such as [PH10, CRS05]. The authors also show how to alter election results using *misprinting attack* with a countermeasure that requires trusting a number of election observers.

Civitas [CCM08] achieves verifiability and coercion-resistance by allowing voters to produce fake voting credentials used for generating ballots that are eventually deleted

with a coercer not being able to detect. This requires the availability of an untappable channel between a voter and one *registration teller*, and Civitas also requires anonymous channel for voters to cast their votes for coercion-resistance.

Kiayias et al. introduced 'DEMOS' in [KZZ15b], the first E2E-V and voter privacy e-voting system in the standard model. Previous E2E-V systems such as [Adi08, ZCC+13] rely on setup assumptions for their E2E-V. However, DEMOS also rely on the existence of a bulletin board and a central election authority, and does not guarantee the *recorded as cast* requirement [CZZ+16].

DEMOS-2 [KZZ15a] was introduced to address the heavy computation and storage overload on the election authority, that arises from the pre-computation of multiple ballot ciphertexts for each voter in an election. In DEMOS-2, voters generate their ballot ciphertexts with associated non-interactive zero-knowledge (NIZK) proofs using a common reference string (CRS) that is produced and proven secure by the election authority.

To address the problem of single points of failure that affects prior e-voting systems [CRS05, PH10, CEC+08, BBE+13, CGS97, CCM08, Nef04, Cha04, KZZ15b], Chondros et al. introduced D-DEMOS [CZZ+16]. For example, a common feature of e-voting systems is to assume the existence of a trusted public bulletin board [MSH17] such as Helios' bulletin board. D-DEMOS is the first distributed end-to-end verifiable e-voting system and it utilises a distributed bulletin board for election process verification. The scheme depends on the honesty of a strong majority of election trustees who are responsible for vote tallying, to guarantee privacy and E2E-V. Further, this scheme relies on a trusted Election Authority (EA) at setup which is subsequently destroyed after it initializes all other system components. Moreover, the scheme, like the other schemes explored, does support delegable voting.

Next, we discuss some delegable voting scheme in the literature. Kulyk et al. in [KMNV16] provides details about e-voting systems that are variants of Helios [CGGI14, CGGI13, KTV15, DvdGdSA12, CGG19] and introduced proxy voting to Helios. This was done by using so-called *delegation credentials* to ensure proxies have

authorisation to vote on behalf of a delegator. However, the scheme requires voters to construct and anonymously send *delegation tokens* to delegates, who submit the tokens and delegated votes. In essence, the delegate casts the vote on behalf of the delegator, and the delegate may cast no vote at all [KMNV16]. Although, the authors provide a back-up delegation option (a delegate assigned a lower priority) in case the main delegate does not cast any vote. However, this implies that the voter has to generate multiple delegation tokens and send to multiple delegates in an anonymised way. In addition to the assumption of anonymous channels between voters and proxies (delegates), the security of the system rely on the trustworthiness of the registration authority.

Zwattendorfer et al. in [ZHT13] proposed a proxy voting (based on liquid democracy) system that relies on physical device (Austrian citizen card and associated eID) to address the security issues with implementations of proxy voting systems. The authors proposed two methods for vote delegation. Namely, **server-based delegation** (offline) where voters encodes the delegation information similar to a ballot and **client-based delegation** (online/manual) where voters copy the ballots of proxies. However, the scheme requires strong assumptions to guarantee the security properties. For example, the reliance on a *Ballot Signer* to authenticate voters constitutes a single point of failure. Further, no formal proofs of security is provided to support the system claims and some of the cryptographic components are unspecified.

The authors in [KNM+17] proposed a coercion-resistant proxy voting scheme which builds on techniques of [JCJ05, CCM08] but requires additional *delegation credential* that a voter, who wants to delegate their voting power, will send to their chosen proxy. The coercion resistant property is claimed from the voter's ability to fake delegation credentials (similar to voting credentials in the original schemes) However, there are no formal proofs of secure delegation nor instantiation of the proposed scheme.

Unlike previous works, we designed and implemented the first provably secure blockchain-based delegable voting scheme (i.e. that supports liquid democracy). We extend the initial scheme to support privacy-preserving voting on blockchains with private stake. These schemes are at the core of our proposed treasury system presented in

Chapter 3 and general decision-making system in Chapter 4 respectively.

2.15 Blockchain voting and Governance

This section provides context on blockchain voting and governance in the academic literature. A number of recent blockchain-based voting systems in the literature such as [AmGMA20,KJ20] mainly discuss and leverage blockchains' distributed architecture as an alternative to a trusted bulletin board. However, many of the claims from cryptographic primitives are not instantiated and security analysis are missing. A survey of blockchain-based e-voting systems in the literature and industry is detailed in [AKW19], and also confirms the gap in literature of a provably secure blockchain-based delegable voting system that supports liquid democracy with end-to-end verifiability.

Khan et al. in [KAK18] leverages the distributed ledger architecture of blockchains and introduced a blockchain-based voting scheme based on [Rya08] and the Multichain[12] blockchain platform. Their proposal mainly leverages the hash of voting transactions to claim verifiability of elections. Moreover, the scheme does not support private proxy voting and provides no formal treatment of the system security properties.

Hjalmarsson et al. in [HHHH18] evaluated the application of blockchain to e-voting and proposed a private permissioned blockchain-based e-voting system that utilises smart contracts. The introduced scheme requires electronic ID authentication.

Corry et al. in [MSH17] proposed *The Open Vote Network*, the first implementation of a self-tallying (allowing any party to compute the tally) e-voting protocol (The Open Vote Network) with voter privacy. The self-tallying procedure is susceptible to *abortive attacks* which is addressed through a fund deposit (lock) and deposit paradigm. The Open Vote network has a safe upper limit of about 50 voters [HHHH18] because all the computations are performed on the smart contract [SGY20].

The need to securely evolve and develop blockchains has caused considerable interest in blockchain governance among the community (of stakeholders), industry and in the literature. Primarily, blockchain governance focuses on *decision rights, accountability and*

[12]https://www.multichain.com/

incentives [LLZ+21] for the evolution (adaptability) and improvement of (the operations of) blockchain. In other words, it covers participants, their powers, roles, rewards (or penalties) and responsibilities. It covers how the different entities collaborate to reach decisions that affects (maintain or change) the blockchain [CDP19]. Blockchain governance covers how community members can provide authoritative decisions on changes to blockchain that require human intervention [MARP20]. Governance is critical for sustainable development and evolution of cryptocurrency systems [FCZ20].

Expectedly, research efforts have been focused on decentralised governance solutions due to the decentralised architecture of blockchains. Some of the issues covered under governance include, decisions of block size, consensus protocol (Proof-of-Work vs Proof-of-Stake), etc.

DiRose et al. in [DM18] analysed the governance process of the Bitcoin and Dash blockchains on the issue of increasing block size using the so-called Governance Analytical Framework of [Huf11]. It investigated why Dash was able to receive approval for increasing block size to 2MB in 24 hours while Bitcoin spent years without reaching a resolution. Similar to the work in [KN+], Mosley et al. in [MPG+20] also analysed blockchain governance through an exploration of the Dash governance system and explored the possibility of collusion among masternodes, hence measure decentralisation in the voting process by constructing a similarity matrix based on voting patterns of the masternodes. The authors identified large clusters of similar voting patterns with more votes than the decentralised majority. The open ballot of the voting system only serves to exacerbate the problem because it makes it easy for a coercer to verify how masternodes voted.

Merrill et al. in [MARP20] proposed a voting scheme for updating blockchain-based protocols that allows *clients* to vote by temporarily locking their tokens. The economic cost to voting is used for gauging support for proposals in a so-called *ping-pong governance model* that involves voting and counter-voting (or voting to veto outcome). This is to allow minor changes pass easily and complex ones eventually pass. However, this can result in an indefinite cycle of voting and counter-voting review periods. Moreover, voting is neither private nor fair because new voters can see partial (current) tallies based on

the choices of other voters, before they submit their own votes.

Baudlet et al. in [BFTK20] proposed the use of masternodes similar to Dash, but allows nodes to reduce locked stake over a period of time, to accommodate other users without much funds (compared to the current masternode requirements). It introduced *Importance score* that can grant 'honorary masternode' status to influential blockchain nodes and enable them participate in governance. However, this proposal does not address the privacy issues of the voting scheme. Moreover, there is no instantiation of the proposed scheme nor the importance score algorithm as confirmed by the authors in the paper.

Furthermore, Fan et al. proposed *Multav* [FCZ20] a multi-chain voting framework for blockchain governance on new blockchains that distributes tokens to stakeholders of other blockchains, to enable them vote on the new blockchain. The main aim is to protect new cryptocurrencies that adopt on-chain governance from attack by wealthy adversaries, due to their small market cap and initial circulating supply.

Zachariadis et al. [ZHS19] provides an exploration of issues surrounding blockchains to enable better understanding of distributed governance of blockchain technology especially as it relates to financial services. Smit et al. [SeMS$^+$20] provides a literature review on decision rights on blockchain governance and proposed a blockchain life-cycle model for better understanding blockchain governance and decision rights. Additionally, Ziolkowski et al. [ZMS20] provides a comprehensive analysis of governance of blockchain systems through identification of decision problems to enable better understanding. Finally, Liu et al. in [LLZ$^+$21] provide a systematic literature review for understanding the state-of-the-art on blockchain governance.

Unlike previous works, the system proposed in Chapter 4 is a useful solution to support the process of reaching decisions that are necessary to evolve and upgrade the blockchain itself, given the established need for blockchains (and software or technology in general) to continually evolve throughout their life-cycles. Moreover, as far as we know, it is the first UC-secure distributed voting scheme for private stake blockchains that support liquid democracy. However, we emphasise that while the solution can be

adapted to other use cases, it is not a solution for general (national/state) governance through blockchains. Rather, it is a tool to support the development or evolution of the blockchain itself.

2.16 Summary

This chapter explored sustainability of funding and fund utilisation for blockchains. It examined different sources of blockchain funding, investigated real-world cryptocurrencies with self-sustenance and community-inclusive development mechanisms, and introduced a categorisation of treasury system models. Within blockchain research and practice, there is increasing interest in treasury systems and increasing adoption among real-world cryptocurrencies. Cryptocurrencies such as Dash, that deploy treasury systems have benefited from its adoption e.g., through community participation in cryptocurrency development, minimising operational centralisation, and avoidance of forks from dispute on development issues etc.

Although treasury systems constitute interesting solution to sustainable blockchain funding, they require careful design and analysis to avoid problems that will hinder their utility. Notably, utmost consideration should be given to source of treasury funds, management of funds, decision-making (e.g., voting rule and systems such as liquid democracy, receipt-free voting), partitionary budgeting, usability, security and privacy, incentives for stakeholders, game-theoretic analysis, cryptocurrency inflation and blockchain (resource) utilisation.

Despite the benefit witnessed from early adoption of variants of the treasury system, these systems experience a number of issues e.g. ad-hoc, lack of incentives, insecure/non-private balloting. Thus, affecting improved adoption and success of pioneering treasury systems. Some of these issues include inadequate design analysis, security, privacy and game-theoretic analysis and 'incentivisation' of the systems prior to development and deployment [KNS+16]. Evidently from the reviewed earlier systems, it is clear that they were created in an ad-hoc fashion and changes were made as the systems mature. That is, the systems are not secure by design and provide little to no analysis of overall security

goals and requirements. Thus, we present our solution, covering these gaps, in Chapter 3.

Further, we provide a description of research work in the literature around e-voting systems, blockchain-based voting and governance. These are related concepts at the core of our proposed blockchain treasury system for cryptocurrencies and the extended privacy-preserving system of Chapter 3 and Chapter 4 respectively. The exploration of the literature establishes the trend from early e-voting systems towards end-to-end verifiable systems, and blockchain-based voting systems. Further, the exploration of literature helps us to establish the research gap of a provably (UC) secure blockchain-based (decentralised) decision-making system that supports delegable voting (liquid democracy). The presentations in the next chapter addresses this gap in literature and also fulfils our second objective from Section 1.2, an important step towards answering the main research questions of this thesis.

Chapter 3

A Collaborative Blockchain Treasury System

This chapter elaborates on the design and development of our provably secure treasury system for cryptocurrencies. The system enables community-inclusive collaborative intelligence gathering and supports incentivised participation and delegative voting (liquid democracy). Liquid democracy, also known as delegative democracy [For02], is a hybrid of direct democracy and representative democracy that aims to provide the benefits of both types of democracy [GA15] by enabling organisations to take advantage of skilled experts in a voting process. Mainly, liquid democracy enhances choice of voters by distributing voting power, assist voters in their decision, improve systems of proportional representation and scale direct democracy through specialisation [For20]. Moreover, it lowers participation cost and optimise attention budgets of participants in a decision-making system.

A treasury system is a community-controlled and decentralized collaborative decision-making mechanism for sustainable funding of blockchain development and maintenance. The Dash governance system (DGS), Zcash [Fou] and Decred [Decb] are real-world examples of treasury systems. For the first time, we provide a rigorous design of the treasury system that is compatible with most existing blockchain infrastructures, such as Bitcoin, Ethereum, etc. because it mainly requires the blockchain to support payment

transactions. At the core of the proposed treasury system is a distributed universally composable secure end-to-end verifiable voting protocol.

The rest of this chapter is organised as follows: Section 3.1 introduces the research presented in the chapter by providing pertinent background, motivates the presentation in this chapter and summarises main contributions. The main contributions are provided in Section 3.2. Section 3.3 discusses relevant cryptographic concepts, tools and abstractions useful to the cryptographic schemes, algorithms and protocols that are used by the treasury system. Section 3.4 details relevant zero-knowledge proofs and constructions required to ensure parties honestly follow the protocol. Section 3.5 discusses the designed treasury system and its features such as entities in the system, incentive mechanism, funding sources, decision-making, etc.

Finally, Section 3.6 details the modeling of the voting scheme adopted in the treasury system and Section 3.7 details the novel unit vector zero knowledge proof with logarithmic communication. Note that, we defer the analysis of security the voting protocol of the treasury system, implementation and performance to Chapter 5.

3.1 Overview

Absence of a centralized control over the operation process is a desirable key feature expected of blockchains in general and cryptocurrencies in particular [AMM18, GKCC14] i.e. blockchain solutions should neither rely on 'trusted parties or powerful minority' for their operations, nor introduce such (centralisation) tendencies into blockchain systems. Decentralization help eliminates *single point of failure* and offers better security guarantees and also enables enhanced user privacy techniques.

Real-world cryptocurrencies require steady funding for continuous development and maintenance of the systems. Accordingly, their maintenance and developmental funding should also be void of centralization. Therefore, secure and 'community-inclusive' long-term sustainability of funding and fund utilisation offers an avenue to fulfil this goal towards ensuring the robustness of blockchains.

In the early years of blockchain development, cryptocurrencies such as Bitcoin,

mainly rely on patron organizations and donations. Recently, an increasing number of cryptocurrencies are funded through *initial coin offering* (ICO) – a popular crowd-funding mechanism to raise money for the corresponding startups or companies.

A major drawback of donations and ICOs is that they lack sustainable funding supply. Consequently, they are not suitable as long-term funding sources for cryptocurrency development due to the difficulty of predicting the amount of funds needed (or that will be available) for future development and maintenance. Alternatively, some cryptocurrency companies, such as *Zcash Electric Coin Company*, take certain percentage of haircut/tax (founders reward) from the miners' reward. This approach would provide the companies a more sustainable funding source for long-term planning of the cryptocurrency development.

Nevertheless, the aforementioned development funding approaches have risks of centralization in terms of decision-making on the development steering. Only a few people (in the organisation or company) participate in the decision-making process on how the available funds will be used. However, the decentralized architecture of blockchain technologies makes it inappropriate to have a centralized control of the funding for secure development processes. Sometimes disagreement among the organisation members may lead to catastrophic consequences. Examples include the splitting of Ethereum to Ethereum and Ethereum Classic and Bitcoin to Bitcoin and Bitcoin cash.

Ideally, all cryptocurrency stakeholders are entitled to participate in the decision-making process on funding allocation. This democratic type of community-inclusive decentralized decision-making enables a better *collaborative intelligence* due to the optimisation of expertise and voter effort and time within the community. The concept of *treasury system* has been raised to facilitate blockchain stakeholder participation in decision-making [KNS+16, Fou, Decb, ZOB19, ZB18].

A treasury system is a community controlled and decentralized collaborative decision-making mechanism for sustainable funding of the underlying blockchain development and maintenance. The core component of a treasury system is a decision-making system that allows members of the community collectively reach some resolution. During each

treasury period, anyone can submit a proposal for projects to be funded. Due to shortage of available funds, only a few of them can be supported. Therefore, a collaborative decision-making mechanism is required.

The Dash governance system [DD14] is a real-world example of of such a system. A treasury system consists of iterative treasury periods. During each treasury period, project proposals are submitted, discussed, and voted for; top-ranked projects are then funded. Earlier forms of treasury systems do not offer ballot privacy to voters [Dasb, KNS+16, Decb, Fou], thereby conceivably adversely impacting soundness of any funding decision due to risk of participant coercion. Moreover, they fail to effectively and efficiently aggregate and optimise expertise within the community in the decision-making process. This is due to the system only supporting basic type of voting schemes and limitation of voting power (thus effective contribution) of experts within the systems. Furthermore, for efficiency, these systems typically allow only a random subset of stakeholders, e.g. Decred Pi, masternodes in DGS, to participate in treasury decision-making process, thereby disenfranchising other community members.

In our treasury system, we put forward a different approach – *liquid democracy* – to achieve better collaborative intelligence. Liquid democracy (also known as delegative democracy [For02]) is a hybrid of direct democracy and representative democracy that enables organisations to take advantage of experts in a treasury voting process by supporting vote delegation, as well as giving the stakeholders the opportunity to directly vote. For each project, a voter can either vote directly or delegate his/her voting power to an expert who is knowledgeable and renowned in the corresponding area. Delegation and ballot is private and voters and experts vote simultaneously, thereby eliminating coercion risks and vote-buying.

In the literature, a few blockchain based e-voting schemes have been proposed [Dasb, Dem17, McC16, nVo17, Ago18, LJEK16]. Some major differences between our treasury system voting scheme and other e-voting schemes include:

- Conventional e-voting scheme requires real-world identity authentication while our treasury decision-making do not need to link voters to their real identities.

- Typically, each voter has one vote, while in our treasury decision-making, the voting power is proportional to the corresponding stake.
- Our treasury decision-making supports liquid democracy with ballot privacy assurance, while, as far as we know, no other e-voting scheme can support liquid democracy with provable security.

Proper selection of the voting scheme allows maximizing the number of voters satisfied by the voting results as well as minimizing voters' effort required to study and adequately analyse the issues being voted on. However, in practice, there are two widely used voting schemes [KN$^+$]:

- *preferential* or *ranked voting*
- *approval voting*

An extension of approval voting is the 'Yes-No-Abstain' voting, where voters express 'Yes/No/Abstain' opinion for each proposal. Recent theoretical analysis of this election rule with variable number of winners, called *Fuzzy threshold voting* [KKN$^+$], shows advantages of this voting scheme for treasury system application. These benefits include simplicity, ease of understanding and result interpretation, election of more moderate candidates [BF78, KKN$^+$]. Furthermore, it guarantees that the Condorcet winner is always a member of the winning set of proposals/candidates. Therefore, we adopt this voting scheme in our treasury system. Nevertheless, we emphasize our treasury system can adopt a different voting scheme with minimal changes to the underlying cryptographic protocols.

Next, we provide a high level description of the basic protocol, covering the core techniques and algorithms used.

3.1.1 Basic Protocol

Here, we provide a high-level description of the stake-weighted delegable voting scheme at the core of our treasury system detailed in Section 3.5. For simplicity, without the

loss of generality, this description covers the main processes in the *pre-voting, voting and post-voting* stages by using an example treasury period with a single proposal.

Pre-voting: At the beginning of the treasury period, a proposer submits a transaction with details of a proposal to support the blockchain. The proposal transaction mainly covers the *an identifier* for the proposal, *the amount of funds* requested, and the *payment address* for the proposal (if successful at the end of voting). Expectedly, these transactions cost some coins to prevent *denial of service attacks*.

After the proposal submission process, intending participants (voters, experts, and voting committee members) in the election process of the treasury system register to participate in the voting process. Voters are stakeholders who deposit stake (corresponding to their voting power) to partake in the election process while experts are a special subset of voters that can receive delegation from other voters in the voting process. The registration transaction mainly covers the participant's *role*, the amount of the participant's *deposited stake*, and the *return address* for payment of voting rewards.

Voting: In the voting stage, the voting committee is selected using a procedure that can be abstracted as follows. Potential committee members combine a seed with their public key and compute a hash of the combination i.e.,

$$\mathsf{digest} = \mathsf{hash}\big(verificationKey, signature(\mathsf{seed})\big)$$

The digest of the hash computation is compared with a system defined threshold T and users are qualified as voting committee members by submitting registration transactions that confirm they are able to satisfy the condition. See Subsection 3.5.4 for a complete description of the voter committee selection. Consequently, the voting committee members jointly generate the election public key to be used for encrypting unit-vector ballots in the election. We adopt the distributed key generation (DKG) protocol of [GJKR99], which uses a (t,n)-threshold verifiable secret sharing and can withstand the corruption of less than half of the committee members. The public key is used to encrypt ballots and the election tally is decrypted using partial decryption shares of the voting committee members. Full details of the protocol is provided in Subsection 3.3.2.

Following voter/expert registration and committee selection, we briefly describe the ballot casting process for voters and experts. We adopt the ElGamal discrete-log problem based cryptosystem due to its homomorphic property.

Specifically, for an election process involving m experts, a voter's ballot is an element-wise encryption of a unit-vector of size $m+3$. Conceptually, this means that a ballot is a length $m+3$ 'concatenation' of individual encryptions of either 0 or 1, where only one position contains 1. In essence, the position of 1 in the individual encryptions indicate whether a ballot is a delegation (positions 1 to m) or a direct vote (positions $m+1$ to *position* $m+3$). The voter ballot is depicted in fig. 3.1 below.

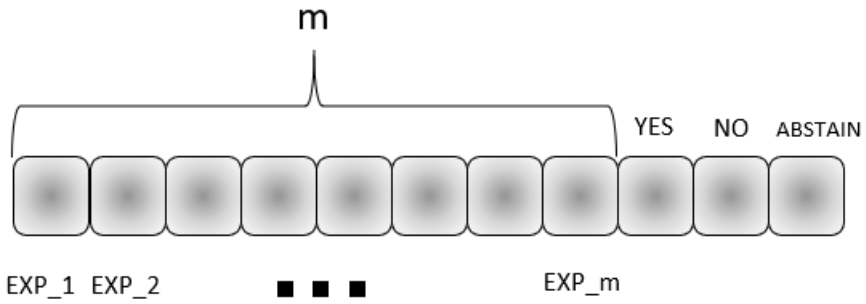

Figure 3.1: A treasury system voter ballot with m experts.

However, the length of unit vector ballots for experts in the voting process is 3. That is, representing a direct vote of *yes, no, or abstain* depending on the position of 1 in the unit vector. To guarantee that ballots of voters and experts alike are formed correctly, they must be accompanied by zero-knowledge proofs (cf. Section 3.7). These proofs are necessary to prevent cheating because ballots are tallied homomorphically, i.e. without first decrypting them. We provide more formal description of the ballot and voting procedures in Subsection 3.6.4.

Post-Voting : At the end of voting, the voting committee members jointly compute the voting tally. Essentially, the tally is a two-step process. In the first step, the delegation section of voters' ballots are parsed and homomorphically added to obtain the (encryption of) total number of delegations received by individual experts as shown in fig. 3.2.

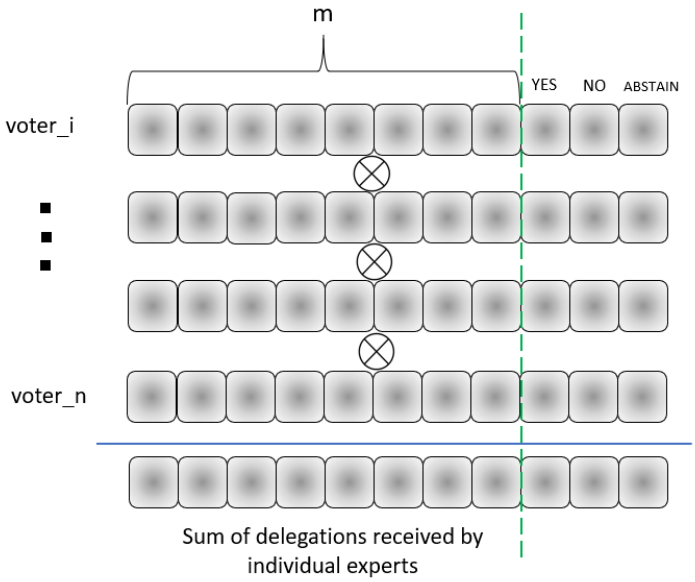

Figure 3.2: First step of the ballot tally process showing the homomorphic addition of the delegations received by individual experts

Following the delegation summation, the voting committee jointly decrypt the delegations ciphertexts to reveal the delegation received by each expert. In the next step, the latter part of voter ballots $(m+1,\ m+2,\ m+3)$ signifying direct voting, is homomorphically added to the ballots the m experts to obtain the encryptions of the tally. Similarly, the voting committee members jointly decrypt the tally to produce the final voting outcome signifying the total number of Yes, No, Abstain votes received by the proposal. Note that before the computing the final tally, each voter ballot is weighted by the voting power of the voter and expert ballot is weighted by the amount of received delegation.

The rest of the presentations in this chapter provides formal and complete description of the entities and processes that make up our proposed treasury system.

3.2 Our Results

In this work, we aim to resolve the funding sustainability issue for long-term cryptocurrency development and maintenance by proposing a novel treasury system. The proposed treasury system is compatible with most existing off-the-shelf cryptocurrencies/blockchain

platforms, such as Bitcoin and Ethereum. We highlight the major contributions of this work as follows.

- For the first time, we provide a detailed security modeling for a blockchain-based treasury voting system that supports liquid democracy/delegative voting. More specifically, we model the voting system in the well-known *Universally Composable* (UC) framework [Can00] via an ideal functionality $\mathcal{F}_{\text{VOTE}}^{t,k,n,m}$. The functionality interacts with a set of voters and experts as well as k voting committee members. It allows the voters to either delegate their voting power to some experts or vote directly on the project. If at least t out of k voting committee members are honest, the functionality guarantees termination. Even in the extreme case, when all the voting committee members are corrupted, the integrity of the voting result is still ensured by the non-interactive zero-knowledge proofs which can be publicly verified by any party; however, in that case we don't guarantee protocol termination.

- We propose an efficient design of the treasury system. The system collects funding via three potential sources: (i) Minting new coins; (ii) Taxation from miners' reward; (iii) Donations or charity. In an iterative process (see Fig. 3.3), the treasury funds accumulate over time, and the projects are funded periodically. Each treasury period consists of pre-voting epoch, voting epoch, and post-voting epoch, which can be defined in terms of number of blockchain blocks. In the pre-voting epoch, project proposals are submitted, and the voters/experts are registered. In the voting epoch, the voting committee is selected; after that, they jointly generate the voting key for the treasury period. The voters and experts then cast their ballots. In the post-voting epoch, the voting committee computes and signs the treasury decision. Winning proposals will then be funded. Any stakeholder in the community can participate in the treasury voting, and their voting power is proportional to their possessed stake. (Without loss of generality, we assume a coin has certain storage field for non-transactional data.)

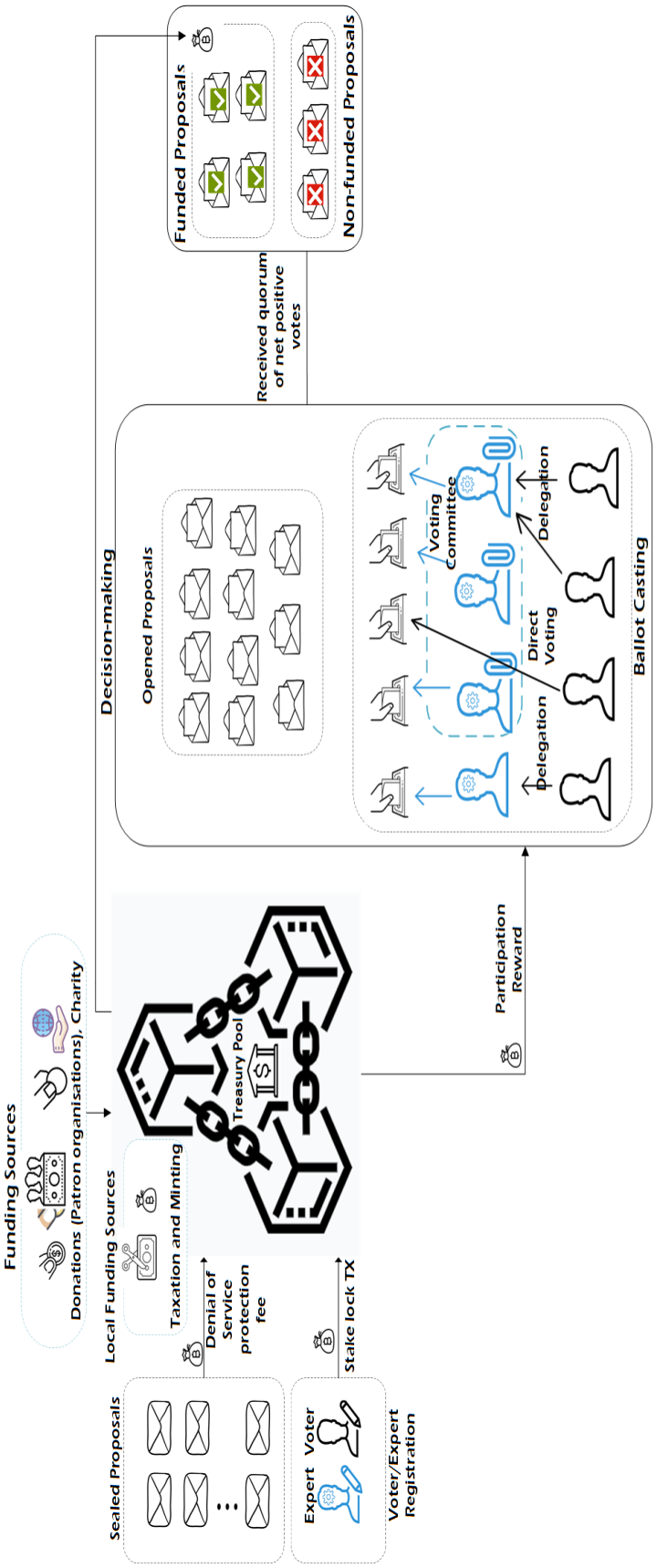

Figure 3.3: A treasury period depicting the parties and processes involved in the treasury system.

- We proposed the world's first honest verifier zero-knowledge proof/argument for unit vector encryption with logarithmic size communication. Conventionally, to show a vector of ElGamal ciphertexts element-wise encrypt a unit vector, Chaum-Pedersen proofs [CP93] are used to show each of the ciphertexts encrypts either 0 or 1 (via Sigma OR composition) and the product of all the ciphertexts encrypts 1. Such kind of proof is used in many well-known voting schemes, e.g., Helios. However, the proof size is linear in the length of the unit vector, and thus the communication overhead is quite significant when the unit vector length becomes larger. In this work, we propose a novel special honest verifier ZK (SHVZK) proof/argument for unit vector that allows the prover to convince the verifier that a vector of ciphertexts $(C_0, \ldots, C_{n-1})$ encrypts a unit vector $\mathbf{e}_i^{(n)}$, $i \in [0, n-1]$ with $O(\log n)$ proof size. The proposed SHVZK protocol can also be Fiat-Shamir transformed to a non-interactive ZK (NIZK) proof in the random oracle model.

- We provide prototype implementation [IOH19] of the proposed treasury system for running and benchmarking in the real world environment. Our implementation is written in Scala programming language over Scorex 2.0 framework and uses TwinsChain consensus for keeping the underlying blockchain. Main functionality includes proposal submission, registration of voters, experts, voting committee members and their corresponding deposit lock, randomized selection of the voting committee members among voters, distributed key generation (6-round protocol), ballots casting, joint decryption with recovery in case of faulty committee members (4-round protocol), randomness generation for the next treasury period (3-round protocol), reward payments, deposit paybacks, and penalties for faulty actors. All implemented protocols are fully decentralized and resilient up to 50% of malicious participants. During verification we launched a testnet that consisted of 12 full nodes successfully operating tens of treasury periods with different parameters.

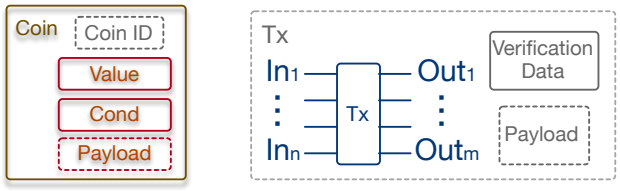

Figure 3.4: Coin and transaction structure.

3.3 Building Blocks

Details of notations adopted in this thesis are presented in Section 2.1. However, here, we provide preparatory information about other key cryptographic abstractions, and building blocks use in the proposed treasury system.

3.3.1 Blockchain Abstraction

Without loss of generality, we abstract relevant underlying cryptocurrency as encompassing the following concepts: *Coin, Address* and *Transaction*. This abstraction helps to simplify our presentation of the treasury system through the use of appropriate concepts that minimise the exploration of inessential details.

◦ *Coin.* We assume the underlying blockchain has the notion of *Coins* or its equivalent. Each coin can be spent only once, and all the value of a coin must be consumed in a spend transaction. Fig. 3.4 depicts four key attributes of each coin explained as follows:

- **Coin ID**: An implicit property of every coin used to uniquely identify the coin from other coins.
- **Value**: Holds the 'amount' of a coin.
- **Cond**: Holds the conditions that must be satisfied for a coin to be correctly/validly spent.
- **Payload**: Holds useful non-transactional data (meta-data for the checks and validations in the treasury system).

◦ *Address.* Conventionally, in blockhain-based cryptocurrencies an address is usually a public key, pk, or hash of a public key, $h(\mathsf{pk})$ [Nak08, NBF+16, KRDO17]. To create

coins associated with an address, the spending condition of the coin is specified as a valid signature under the corresponding public key pk of the address. For the purpose of the treasury system, an address is defined as a generic representation of some spending condition of a coin. Using a recipient's address, a sender is able to create a new coin whose spending condition is one that only the recipient can satisfy, based on the public information of the recipient's address. Thus, the recipient may legitimately spend the coin later in a valid transaction as specified by our blockchain abstraction.

◦ *Transaction.* Every transaction takes one or more unspent coins, denoted by $\{\mathsf{In}_i\}_{i\in[n]}$, as input, and outputs one or more new coins, denoted by $\{\mathsf{Out}_j\}_{j\in[m]}$. Except for special transactions (e.g. specific treasury system transactions), the following condition holds for every transaction:

$$\sum_{i=1}^{n} \mathsf{In}_i.\mathsf{Value} \geq \sum_{j=1}^{m} \mathsf{Out}_j.\mathsf{Value}$$

and the difference in value between the input and output coins is construed as transaction fee. That is,

$$\mathsf{Tx}_{\mathsf{fee}} = \sum_{i=1}^{n} \mathsf{In}_i.\mathsf{Value} - \sum_{j=1}^{m} \mathsf{Out}_j.\mathsf{Value}$$

As shown in Fig. 3.4, the transaction has a Verification Data field that contains pertinent verification data to satisfy all the spending conditions of the input coins $\{\mathsf{In}_i\}_{i\in[n]}$. An example of verification data is valid signatures from all the owners of the input coins.

In addition, each transaction also has a Payload field that can be used to store any additional non-transactional data. Formally, a transaction is denoted as:

$$\mathsf{Tx}(A; B; C)$$

where A is the set of input coins, B is the set of output coins, and C is the Payload field. We remark that to the simplify notation, the verification data in a transaction is implicit (not explicitly declared).

3.3.2 Distributed Key Generation

In this section, we give a full description of our distributed key generation (DKG) protocol. A secure DKG is essential and a fundamental building block in the voting protocol within our treasury system. This is because the underlying blockchain does not have a 'single trusted third-party' that can independently generate the public key credentials required for securing the ballots (of participants) and other cryptographic operations within the treasury.

Naively, threshold distributed key generation can be achieved in the following way. Every party $\mathcal{P}_i$, $i \in [n]$ to the DKG protocol initially generates a public/private key pair $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{param})$. Each $\mathcal{P}_i$ then posts pk_i publicly (on the blockchain or bulletin board) and use $(t+1, n)$-threshold *verifiable secret sharing* (VSS) to share sk_i to all the other committee members. The combined voting public key can then be constructed as $\mathsf{pk} := \prod_{i=1}^{n} \mathsf{pk}_i$.

However, this approach is insecure because an adversary can influence the distribution of the final voting public key pk by allowing the corrupted parties to selectively abort the protocol execution. A detailed discussion of this attack scenario is covered in [GJKR99].

We adopt the distributed key generation protocol proposed by Gennaro *et al.* [GJKR99] that addresses the subtle attack described earlier. Mainly, the DKG protocol allows parties $\mathcal{P}_i$ to initially post a 'commitment' of pk_i. The committed pk_i for every party is then revealed by $\mathcal{P}_i$ after sharing of the corresponding sk_i through $(t+1, n)$-threshold VSS. This helps to prevent malicious parties from influencing the distribution of the generated key.

Specifically, our DKG is an adaptation of the DKG proposed by Genarro *et al.* which allows accommodation of up to $t < n/2$ malicious parties in the protocol. Thereby, guaranteeing that, with at least $\lfloor \frac{n}{2} \rfloor + 1$ honest parties, all players should be able to agree on a uniformly random public key pk such that no malicious party (parties) can influence the distribution of the generated public key. The corresponding secret key sk is shared among all the parties. Ideally, protocol termination should be guaranteed when up to $t = \lceil \frac{n}{2} \rceil - 1$ out of n committee members are corrupted.

Our protocols use the blockchain to realise the broadcast channel and peer-to-peer channels. Next is a detailed description of our distributed key generation protocol.

3.3.2.1 The Treasury System Distributed Key Generation

Given (g, h) as the Common Reference String (CRS), let $\mathsf{C} := \{\mathsf{C}_1, \mathsf{C}_2, ..., \mathsf{C}_k\}$ be the set of election committee members, and let pk_i be the public key associated with C_i, for $i \in [k]$. The adversary is able to corrupt up to $t < k/2$ committee members.

In the first round, each committee member C_i selects a random $(a_{i,0}, ...a_{i,t})$ and random $(a'_{i,0}, ..., a'_{i,t})$ where t is maximum number of members that can be corrupted. Each member then define two polynomials of degree t in the following form:

$$f_i(x) = a_{i,0} + a_{i,1}x + ... + a_{i,t}x^t$$

and

$$f'_i(x) = a'_{i,0} + a'_{i,1}x + ... + a'_{i,t}x^t$$

Therefore, each committee member C_i contributes $x_i = a_{i,0} = f(0)$ to the combined secret $x = \mathsf{sk}$. Furthermore, to confirm the correctness of commitments, each member C_i posts a corresponding commitment $E_{i,l} = g^{a_{i,l}} h^{a'_{i,l}}$ on the blockchain. For every other member of the election committee, each C_i computes $s_{i,j} = f_i(j)$ and $s'_{i,j} = f'_i(j)$ and posts the encryption of $s_{i,j}$ and $s'_{i,j}$ under the public key of j to the blockchain. Note that, $j \neq i$, i.e. each member posts $e_{i,j} \leftarrow \mathsf{Enc}_{pk_j}(s_{i,j})$ and $e'_{i,j} \leftarrow \mathsf{Enc}_{pk_j}(s'_{i,j})$. Only C_j can decrypt these commitments and this signifies the end of the first round.

In the second round, each committee member C_j fetch all $e_{i,j}$ and $e'_{i,j}$ encrypted under their public key from the blockchain and decrypt them using their private key sk_i to obtain their corresponding shares $s_{i,j}$ and $s'_{i,j}$. In order to verify that the shares they have received are valid and not in error, each committee member checks if: $g^{s_{i,j}} h^{s'_{i,j}} = \prod_{l=0}^{t} (E_{i,l})^{i^l}$ for $i \in [k],\ i \neq j$. Where this check fails, C_j posts a complain against C_i by revealing the evidence:

$$(s_{i,j}, s'_{i,j}) \qquad \text{and}$$

$$\begin{aligned}\pi \leftarrow \mathsf{NIZK}\big\{(s_{j,i}, s'_{j,i}, \mathsf{pk}_i, e_{j,i}, e'_{j,i}), (\mathsf{sk}_i) :\\ s_{j,i} = \mathsf{Dec}_{\mathsf{sk}_i}(e_{j,i})\\ \wedge\ s'_{j,i} = \mathsf{Dec}_{\mathsf{sk}_i}(e'_{j,i})\\ \wedge\ (\mathsf{pk}_i, \mathsf{sk}_i) \in \mathcal{R}_{\mathrm{PKE}}\big\}.\end{aligned}$$

Members with one valid complain against them are excluded from participating in the key generation process.

In the third round, following the exclusion of no member or some members based on the outcome of round 2, each qualified committee member C_i posts $A_{i,\ell} := g^{a_{i,\ell}}$ for $\ell \in [\mathcal{J}]$ to the blockchain. Each committee member also return its secret key share as

$$\overline{\mathsf{sk}}_i := \sum_{j \in [\mathcal{J}]} \gamma_i \cdot s_{j,i}$$

where $\gamma_i := \prod_{\ell \in \mathcal{J} \setminus \{i\}} \frac{\ell}{\ell - i}$.

In the fourth round, each qualified committee member C_i, for $j \in \mathcal{J}, j \neq i$, checks if :

$$g^{s_{j,i}} = \prod_{\ell=0}^{t} (A_{j,\ell})^{i^\ell}.$$

Where the check fails, C_i posts complain against C_j together with the evidence $(s_{j,i}, s'_{j,i})$ on the blockchain, such that:

$$g^{s_{j,i}} h^{s'_{j,i}} = \prod_{\ell=0}^{t} (E_{j,\ell})^{i^\ell} \qquad \text{and}$$

$$g^{s_{j,i}} \neq \prod_{\ell=0}^{t} (A_{j,\ell})^{i^\ell}$$

In the fifth and final round, each qualified committee member C_i checks if complaints raised against qualified committee members in the fourth round are valid. For all members against whom valid complaints were raised, other qualified committee members posts the shares $s_{j,i}$ they received from the members who have complaints against them. Thereby,

enabling the reconstruction of the secret share of the erring qualified committee members as follows:

$$\mathsf{sk}_j := \sum_{i\in[\mathcal{J}]} \gamma_j \cdot s_{i,j} \quad \text{and redefine} \quad A_{j,0} := g^{\mathsf{sk}_j},$$

where $\gamma_j := \prod_{\ell\in\mathcal{J}\setminus\{j\}} \frac{\ell}{\ell-j}$.

Finally, the combined (election) public key is computed as follows:

$$\mathsf{pk} := \prod_{j\in[\mathcal{J}]} A_{j,0}$$

Distributed key generation $\Pi_{\mathbf{DKG}}$

Round 1: Each committee member C_i do the following:

- Pick random $a_{i,0}, a_{i,1}, \ldots, a_{i,t}, b_{i,0}, b_{i,1}, \ldots, b_{i,t} \leftarrow \mathbb{Z}_p$.
- Define two polynomials $f_i(x) := \sum_{\ell=0}^{t} a_{i,\ell} x^\ell$ and $f'_i(x) := \sum_{\ell=0}^{t} b_{i,\ell} x^\ell$.
- For $\ell \in [t]$, post $E_{i,\ell} := g^{a_{i,\ell}} h^{b_{i,\ell}}$ on the blockchain.
- For every other C_j, $j \in [k], j \neq i$, compute $s_{i,j} := f_i(j)$ and $s'_{i,j} := f'_i(j)$ and post $e_{i,j} \leftarrow \mathsf{Enc}_{\mathsf{pk}_j}(s_{i,j})$ and $e'_{i,j} \leftarrow \mathsf{Enc}_{\mathsf{pk}_j}(s'_{i,j})$ on the blockchain. (Only C_j can decrypt them.)

Round 2: Each committee member C_i do the following:

- Fetch $\{(e_{j,i}, e'_{j,i})\}_{j\in[k],j\neq i}$ from the blockchain, and use pk_i to decrypt them, obtaining the corresponding shares $\{(s_{j,i}, s'_{j,i})\}_{j\in[k],j\neq i}$.
- For $j \in [k], j \neq i$, check if $g^{s_{j,i}} h^{s'_{j,i}} = \prod_{\ell=0}^{t} (E_{j,\ell})^{i^\ell}$. If not, post complain against C_j by revealing the evidence: $(s_{j,i}, s'_{j,i})$ and

$$\begin{aligned} \pi \leftarrow \mathsf{NIZK}\big\{(s_{j,i}, s'_{j,i}, \mathsf{pk}_i, e_{j,i}, e'_{j,i}), (\mathsf{sk}_i) : \\ s_{j,i} = \mathsf{Dec}_{\mathsf{sk}_i}(e_{j,i}) \\ \wedge\ s'_{j,i} = \mathsf{Dec}_{\mathsf{sk}_i}(e'_{j,i}) \\ \wedge\ (\mathsf{pk}_i, \mathsf{sk}_i) \in \mathcal{R}_{\mathrm{PKE}}\big\}. \end{aligned}$$

- (One valid complain against C_j, $j \in [k]$ will disqualify C_j.)

Round 3: Define the indices of the qualified set of committee members as $\mathcal{J}$. Each committee member C_i do the following:

- For $\ell \in [t]$, post $A_{i,\ell} := g^{a_{i,\ell}}$ to the blockchain.
- Return its secret key share as $\mathsf{sk}_i := \sum_{j\in[\mathcal{J}]} \gamma_i \cdot s_{j,i}$, where $\gamma_i := \prod_{\ell\in\mathcal{J}\setminus\{i\}} \frac{\ell}{\ell-i}$.

Round 4: Each committee member C_i do the following:

- For $j \in \mathcal{J}, j \neq i$, check if $g^{s_{j,i}} = \prod_{\ell=0}^{t}(A_{j,\ell})^{i^\ell}$. If not, post complain against C_j together with the evidence $(s_{j,i}, s'_{j,i})$ on the blockchain. Such that $g^{s_{j,i}} h^{s'_{j,i}} = \prod_{\ell=0}^{t}(E_{j,\ell})^{i^\ell}$ and $g^{s_{j,i}} \neq \prod_{\ell=0}^{t}(A_{j,\ell})^{i^\ell}$.

Round 5: Each committee member C_i do the following:

- If there is a valid complain against C_j, $j \in \mathcal{J}$, then post the share $s_{j,i}$ on the blockchain. (Everyone can reconstruct $\mathsf{sk}_j := \sum_{i\in[\mathcal{J}]} \gamma_j \cdot s_{i,j}$ and re-define $A_{j,0} := g_j^{\mathsf{sk}}$, where $\gamma_j := \prod_{\ell\in\mathcal{J}\setminus\{j\}} \frac{\ell}{\ell-j}$.)
- Return the election public key as $\mathsf{pk} := \prod_{j\in[\mathcal{J}]} A_{j,0}$.

Figure 3.5: Distributed key generation Π_{DKG}

3.3.3 UC Ideal Functionalities

In the following, we summarise UC ideal functionalities as building blocks adopted in this research work.

3.3.3.1 The Distributed Key Generation Functionality

We use the key generation functionality $\mathcal{F}_{\mathrm{DKG}}^{t,k}$ of [Wik04] for threshold key generation of the underlying public key cryptosystem. The functionality is depicted in Fig. 3.6 and it interacts with a set of committees $\mathcal{C} := \{\mathsf{C}_1, \ldots, \mathsf{C}_k\}$ to generate a public key pk and

deal the corresponding secret key sk among the committees. In Section 3.3.2, we provide a detailed description of our threshold distributed key generation protocol adopted from Gennaro *et al.* [GJKR99].

Functionality $\mathcal{F}_{\text{DKG}}^{t,k}$

The functionality $\mathcal{F}_{\text{DKG}}^{t,k}$ interacts with a set of voting committees $\mathcal{C} := \{\mathsf{C}_1, \ldots, \mathsf{C}_k\}$, any party $\mathcal{P}$, and the adversary $\mathcal{A}$. It is parameterised with variable $\mathcal{J}$ and algorithms $\mathsf{KeyGen}()$ and $\mathsf{Deal}()$.
Initially, set $\mathcal{J} := \emptyset$.

- Upon receiving $(\textsc{KeyGen}, \mathsf{sid})$ from $\mathsf{C}_j \in \mathcal{C}$, set $\mathcal{J} := \mathcal{J} \cup \{\mathsf{C}_j\}$.
- If $|\mathcal{J}| = k$, generate a public key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and compute $(\mathsf{sk}_1, \ldots, \mathsf{sk}_k) \leftarrow \mathsf{Deal}(t, k, \mathsf{sk})$. For $j \in [k]$, send message $(\textsc{PrivKey}, \mathsf{sid}, \mathsf{sk}_j)$ to C_j.
- Upon receiving $(\textsc{ReadPK}, \mathsf{sid})$ from any party $p \in \mathcal{P}$, if pk is not defined yet, ignore the request. Otherwise, it sends $(\textsc{PublicKey}, \mathsf{sid}, \mathsf{pk})$ to the requestor.

Figure 3.6: Functionality $\mathcal{F}_{\text{DKG}}^{t,k}$

3.3.3.2 The Global Clock Functionality.

The global clock functionality $\mathcal{G}_{\text{CLOCK}}$ interacts with all the parties. To accommodate offline parties, parties can register and deregister themselves to the functionality $\mathcal{G}_{\text{CLOCK}}$. The clock advances if and only if all registered honest parties have sent TICK command to it. The functionality is depicted in detail in Fig. 3.7.

3.3.3.3 The Ledger Ideal Functionality

Our protocol builds on state-of-the-art ledger ideal functionality proposed by Badertscher et al., [BMTZ17]. As shown, in Fig. 3.11, the functionality maintains a set of registered parties $\mathbb{P}$, a (sub-)set of honest parties $\mathcal{H} \subseteq \mathbb{P}$, and a (sub-set) of de-synchronized honest parties $\mathbb{P}_{DS} \subset \mathcal{H}$. The set $\mathbb{P}, \mathbb{P}_{DS}, \mathcal{H}$ are initially set to $\emptyset$. When a new honest party is registered, it is added to $\mathbb{P}_{DS}$ (hence also to $\mathcal{H}$ and $\mathbb{P}$) and the current time of registration

Functionality $\mathcal{G}_{\text{CLOCK}}$

The functionality interacts with a set of parties $\mathbb{P}$, a set of functionalities $\mathbb{F}$, and the adversary $\mathcal{A}$. It is parametrized with variable τ, $\mathbb{P}$, and $\mathbb{F}$.
Initially, set $\tau := 0$, $\mathbb{P} := \emptyset$, and $\mathbb{F} := \emptyset$.

Registration:

- Upon receiving (REGISTER, sid) from party p, set $\mathbb{P} := \mathbb{P} \cup \{p\}$ and create variable $T_p := 0$.
- Upon receiving (REGISTER, sid) from functionality $\mathcal{F}$, set $\mathbb{F} := \mathbb{F} \cup \{\mathcal{F}\}$ and create variable $T_{\mathcal{F}} := 0$.
- Upon receiving (DE-REGISTER, sid) from party p, set $\mathbb{P} := \mathbb{P} \setminus \{p\}$ and remove variable T_p.
- Upon receiving (DE-REGISTER, sid) from functionality $\mathcal{F}$, set $\mathbb{F} := \mathbb{F} \setminus \{\mathcal{F}\}$ and remove variable $T_{\mathcal{F}}$.
- Upon receiving (GET-REG, sid) from $\mathcal{A}$, return (GET-REG, sid, $\mathbb{P}, \mathbb{F}$) to $\mathcal{A}$.

Synchronization:

- Upon receiving (TICK, sid) from party $p \in \mathbb{P}$, set $T_p := 1$; Invoke procedure Clock-Update and send (TICK, sid, p) to $\mathcal{A}$.
- Upon receiving (TICK, sid) from functionality $\mathcal{F} \in \mathbb{F}$, set $T_{\mathcal{F}} := 1$; Invoke procedure Clock-Update and send (TICK, sid, $\mathcal{F}$) to $\mathcal{F}$.
- Upon receiving (GETTIME, sid) from any participant, return (GETTIME, sid, τ) to the requester.

Procedure Clock-Update:

- If $T_{\mathcal{F}} = 1$ for all $\mathcal{F} \in \mathbb{F}$ and $T_p = 1$ for all the honest $p \in \mathbb{P}$, then set $\tau := \tau + 1$, and reset $T_{\mathcal{F}} := 0$ for all $\mathcal{F} \in \mathbb{F}$ and $T_p := 0$ for all $p \in \mathbb{P}$.

Figure 3.7: Functionality $\mathcal{G}_{\text{CLOCK}}$

is recorded. Similarly, when a party is deregistered, it is removed from both $\mathbb{P}$ and $\mathbb{P}_{DS}$. For each party $p \in \mathbb{P}$, the functionality maintains a pointer $\mathsf{pt_i}$ (initially set to 1) and a current state view $\mathsf{state_i} := \epsilon$ (initially set to empty). The functionality also keeps track of the timed honest-input sequence in a vector $\mathsf{I_H^T}$ (initially $\mathsf{I_H^T} := \epsilon$).

Functionality $\mathcal{F}_{\mathbf{Ledger}}$

It is parametrized by four algorithms $\mathsf{Validate}, \mathsf{ExtendPolicy}, \mathsf{Blockify}$, and $\mathsf{predict\text{-}time}$, along with two parameters: $\mathsf{windowSize}, \mathsf{Delay} \in \mathbb{N}$. The functionality manages variables $\mathsf{state}, \mathsf{NxtBC}, \mathsf{buffer}, \tau_L$ and τ_{state}.

Initially, $\mathsf{state} := \tau_{\mathbf{state}} := \mathsf{NxtBC} := \epsilon, \mathsf{buffer} := \emptyset, \tau_{\mathsf{L}} = 1$.

The functionality maintains the set of registered parties $\mathbb{P}$, the (sub-)set of honest parties $\mathcal{H} \subseteq \mathbb{P}$, and the (sub-set) of de-synchronized honest parties $\mathbb{P}_{DS} \subset \mathcal{H}$. The set $\mathbb{P}, \mathbb{P}_{DS}, \mathcal{H}$ are all initially set to $\emptyset$. When a new honest party is registered, it is added to all $\mathbb{P}_{DS}$ (hence also to $\mathcal{H}$ and $\mathbb{P}$ and the current time of registration is also recorded; similarly, when a party is deregistered, it is removed from both $\mathbb{P}$ and $\mathbb{P}_{DS}$. For each party $p \in \mathbb{P}$, the functionality maintains a pointer $\mathsf{pt_i}$ (initially set to 1) and a current state view $\mathsf{state_i} := \epsilon$ (initially set to empty). The functionality also keeps track of the timed honest-input sequence in a vector $\mathsf{I_H^T}$ (initially $\mathsf{I_H^T} := \epsilon$)

Upon receiving any input I from any party or from the adversary, send $(\textsc{GetTime}, \mathsf{sid})$ to $\mathcal{G}_{\textsc{Clock}}$ and upon receiving response $(\textsc{GetTime}, \mathsf{sid}, \tau)$ set $\tau_L := \tau$ and do the following:

- Let $\hat{\mathbb{P}} \subseteq \mathbb{P}_{DS}$ denote the set of desynchronized honest parties that were registered at time $\tau' \leq \tau_L - \mathsf{Delay}$. Set $\mathbb{P}_{DS} := \mathbb{P}_{DS} \setminus \mathbb{P}$.
- If I was received from an honest party $p \in \mathbb{P}$:
  - Set $\mathbf{I_H^T} := \mathbf{I_H^T} || (I, p, \tau_L)$;
  - Compute $\mathbf{N} = (\mathbf{N_1}, \ldots, \mathbf{N_l}) :=$ $\mathsf{ExtendPolicy}(\mathbf{I_H^T}, \mathsf{state}, \mathsf{NxtBC}, \mathsf{buffer}, \tau_{\mathbf{state}})$ and if $\mathbf{N} \neq \epsilon$, set $\mathsf{state} := \mathsf{state} || \mathsf{Blockify}((\mathbf{N_l 1})) || \ldots, \mathsf{Blockify}((\mathbf{N_l}))$ and $\tau_{\mathbf{state}} := \tau_{\mathbf{state}} || \tau_L^l$, where $\tau_L^l = \tau_L || , \ldots \tau_L$

- For each $\mathsf{BTX} \in \mathsf{buffer}$: if $(\mathsf{Validate}, \mathsf{BTX}, \mathsf{state}, \mathsf{buffer}) = 0$ then delete BTX from buffer. Also reset $\mathsf{NxtBC} := \epsilon$
- If there exists $p_j \in \mathcal{H}$ such that $|\mathsf{state}| - pt_j > \mathsf{windowSize}$ or $pt_j < |\mathsf{state}|$, then set $pt_k := |\mathsf{state}|$ for all $p_k \in \mathcal{H} \setminus \mathbb{P}_{DS}$

- Depending on the above input I and its sender's ID, $\mathcal{F}_{\text{LEDGER}}$ executes the corresponding code from the following list:
  - Submitting a transaction:

    If $I = (\text{SUBMIT}, sid, tx)$ and is received from a party $p \in \mathbb{P}$ or from $\mathcal{A}$ (on behalf of a corrupted party p) do the following
    - Choose a unique transaction ID txid and set $\mathsf{BTX} := (tx, txid, \tau_L, p_i)$
    - if $\mathsf{Validate}(\mathsf{BTX}), \mathsf{state}, \mathsf{buffer} = 1$, then $\mathsf{buffer} := \mathsf{buffer} \cup \{\mathsf{BTX}\}$.
    - Send $(\text{SUBMIT}, \mathsf{BTX})$ to $\mathcal{A}$
  - Reading the state:

    If $I = (\text{READ}, sid)$ is received from a party $p \in \mathbb{P}$ then set $\mathsf{state}|_{\mathsf{min}\{\mathsf{pt_i}, |\mathsf{state}|\}}$ and return $(\text{READ}, sid, \mathsf{state_i})$ to the requester. If the requester is $\mathcal{A}$ then send $(\mathsf{state}, \mathsf{buffer}, \mathbf{I_H^T})$ to $\mathcal{A}$
  - Maintaining the ledger state:

    If $I = (\text{MAINTAIN-LEDGER}, sid, minerID)$ is received by an honest party p in $\mathbb{P}$ and (after updating $\mathbf{I_H^T}$ as above) $\mathsf{predcit\text{-}time}(\mathbf{I_H^T}) = \hat{\tau} > \tau_L$ then send $(\text{TICK}, \mathsf{sid})$ to $\mathcal{G}_{\text{CLOCK}}$. Else, send I to $\mathcal{A}$.
  - The adversary proposing the next block:

    If $I = (\text{NEXT-BLOCK}, \mathsf{hFlag}, (txid_1, \ldots, txid_l))$ is sent from the adversary, update NxtBC as follows:
    - Set $\mathsf{listOfTxid} \leftarrow \epsilon$
    - For $i = 1, \ldots, l$ do: if there exists $\mathsf{BTX} := (x, txid, minerID, \tau_L, p_i) \in \mathsf{buffer}$ with ID $txid = txid_i$ then

set $\mathsf{lisOfTxid} := \mathsf{listOfTxid}||txid_i$

* Finally, set $\mathsf{NxtBC} := \mathsf{NxtBC}||(hFlag, listOfTxid)$ and output (NEXT-BLOCK, ok) to $\mathcal{A}$

– The adversary setting state-slackness:

If $I = (\text{SET-SLACK}, (p_{i1}, \hat{pt_{i1}}), (p_{il}, \hat{pt_{il}}),)$ with $\{p_{i1}, \ldots, p_{il}\} \subseteq \mathcal{H} \setminus \mathbb{P}_{DS}$ is received from the adversary $\mathcal{A}$ do the following:

* If for all $j \in [l] : |\mathsf{state}| - \hat{pt_{ij}} \geq \mathsf{state_{ij}}|, setpt_i := \hat{pt_i}$ for every $j \in [l]$ and return (SET-SLACK, ok) to $\mathcal{A}$.

* Otherwise set $pt_j := \mathsf{state}$ for all $j \in [l]$

– The adversary setting the state for desynchronised parties:

If $I = (\text{DEESYNC-STATE}, (p_{i1}, \mathsf{state'_{il}}))$ with $\{p_{i1}, \ldots, p_{il}\} \subseteq \mathbb{P}_{DS}$ is received from the adversary $\mathcal{A}$, set $\mathsf{state_{ij}} := \mathsf{state'_{ij}}$ for each $j \in [l]$ and return (DESYNC-STATE, ok) to $\mathcal{A}$.

Figure 3.8: Functionality $\mathcal{F}_{\text{LEDGER}}$

3.4 Non-interactive Zero-knowledge Protocols

Looking ahead, in this section we cover non-interactive zero-knowledge proofs protocol useful for convincing other participants of the voting process in the treasury system of one's correct operation and following of the protocol. These pertinent zero-knowledge proof protocols used in the treasury system are available in Appendix A.

3.5 The Proposed Treasury System

The proposed blockchain treasury system is characterised by iterative treasury periods during which all of the treasury processes are repeated in every period. Conceptually, every treasury period is divided into three epochs. Namely :

- Pre-voting epoch
  - Project proposing stage
  - Voter/expert registration stage
- Voting epoch
  - Committee selection stage
  - Key setup stage
  - Ballot casting stage
- Post-voting epoch
  - Tally stage
  - Execution stage

Figure 3.9 depicts a treasury period, the three epochs and their attendant stages. The pre-voting epoch consists of two concurrent stages: project proposing stage and voter/expert registration stage. In the project proposing stage, users can submit project proposals (to support cryptocurrency development) requesting funds from the treasury. At the same time, interested stakeholders can register themselves as either voters and/or experts to participate in the decision making process by locking certain amounts of their stake in the underlying cryptocurrency. A voter's voting power is proportional to his locked stake. For experts, their voting power is proportional to the amount of voting (power) delegation they receive. Delegation is covered in more details in Section 3.5.6. Similarly, a voter's (respectively expert's) treasury reward is proportional to his locked stake (respectively his received delegations).

At the beginning of the voting epoch, there is a voting committee selection stage, where a set of voting committee members are randomly selected from all registered voters who are willing to be considered for selection to the committee. For any voter, the probability of being selected to the committee is proportional to the amount of locked stake. Following the selection of the voting committee members, they jointly

run the distributed key generation protocol to setup the election public key. Thereafter, voters and experts can submit their (encrypted) ballots in the ballot casting stage. Cast ballots are either direct or indirect votes on the proposals, i.e. voters can either specify their choice or delegate their voting powers to some expert. Voting is on a per-project basis, therefore within a given treasury period, voters can delegate their voting power to different experts for different projects, vote directly, or use a combination of both options for different proposals.

In the post-voting epoch, the voting committee members jointly compute and announce the tally result on the blockchain in the tally stage. Finally, in the execution stage, the winning projects are funded, and voters, experts and voting committee members are rewarded (or punished based on their actions in the treasury period) accordingly. These transactions will be jointly signed and executed by the voting committee. Meanwhile, the committee members also jointly commit to a random seed, which is used to select a new voting committee for the next treasury period.

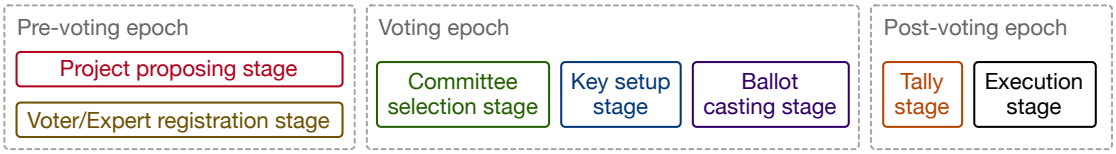

Figure 3.9: Treasury system epochs.

In the following, we elaborate on the principal entities that constitute the treasury system.

3.5.1 Entities

At the core of our treasury system exists a collaborative decision-making system that enables all stakeholders to actively participate and make meaningful contributions towards cryptocurrency development. Let k, ℓ, n, m be integers in poly(λ). The stakeholders may have one or more of the following roles.

- The project owners $\mathcal{O} := \{\mathsf{O}_1, \ldots, \mathsf{O}_k\}$ are a set of stakeholders (or general community members) that have submitted project proposals for supporting the blockchain.

- The voting committees $\mathcal{C} := \{\mathsf{C}_1, \ldots, \mathsf{C}_\ell\}$ are a set of stakeholders that are responsible for generating the voting public key and announcing the voting result.
- The voters $\mathcal{V} := \{\mathsf{V}_1, \ldots, \mathsf{V}_n\}$ are a set of stakeholders that lock certain amount of stake to participate in the treasury system voting on proposals.
- The experts $\mathcal{E} := \{\mathsf{E}_1, \ldots, \mathsf{E}_m\}$ are a special type of voters with specialist knowledge and expertise in some field e.g. blockchain, cryptocurrency, cryptography, economics.

3.5.2 Project Proposal

We adopt a two-stage project proposal submission scheme to prevent granting unfair advantage to late proposal submissions (who might have seen early submissions). To achieve this, we also use the well-known 'commit and open later' approach.

In the first stage, project owners $\mathsf{O}_1, \ldots, \mathsf{O}_k$ post an encryption of their project proposals, encrypted under the election public key of the previous treasury period, to the blockchain. At the end of pre-voting epoch and at the beginning of the voting epoch, the voting committee of previous treasury period will jointly decrypt the 'committed' projected proposals submitted at the pre-voting epoch. In addition to decrypting the proposals, they also reveal a seed, seed (cf. Section 3.5.9).

To commit a project, the project owner needs to submit a special transaction in the following form:

$$\mathsf{Tx}\Big(\{\mathsf{In}_i\}_{i=1}^{n}; \mathsf{TCoin}; \{\textsc{Project}, \mathsf{TID}, \mathsf{P\text{-}Enc}, \mathsf{Addr}\}\Big) \ ,$$

where $\{\mathsf{In}_i\}_{i=1}^{n}$ are the input coins, and TCoin is a special output coin whose spending condition is specified as : the coin can only be spent according to the corresponding treasury decision (cf. Section 3.5.5). Moreover, for the coin value, the following condition must hold:

$$\mathsf{TCoin.Value} \geq \alpha_{\min} \ ,$$

where $\alpha_{\min}$ is the minimum required fee for a project proposal submission, to prevent

denial-of-service attacks. In the transaction Payload field, PROJECT is a tag that indicates it is a special project proposal transaction; TID is the treasury ID that is used to uniquely identify a treasury period; P-Enc is the encrypted project proposal, and Addr is the return address for the project owner to receive funds if the project is successful in the decision-making (voting) process.

3.5.3 Voter/Expert Registration

In order to register to be a voter, a stakeholder (or a set of stakeholders combining their stakes) need(s) to submit a special *voter registration transaction* in the following form to the blockchain:

$$\mathsf{Tx}\Big(\{\mathsf{In}_i\}_{i=1}^{n}; \mathsf{TCoin}; \Big\{\textsc{Voter-Reg}, \mathsf{TID}, \{\mathsf{S}_i\}_{i=1}^{\ell}, \mathsf{S\text{-}Cond}, \mathsf{vk}, \mathsf{Addr}\Big\}\Big) \ ,$$

where $\{\mathsf{In}_i\}_{i=1}^{n}$ are the input coins, and TCoin is a special output coin whose spending condition is defined in Section 3.5.5. In the Payload field, VOTER-REG is a tag that indicates it is a special voter registration transaction; TID is the treasury ID that is used to uniquely identify a treasury period; $\{\mathsf{S}_i\}_{i=1}^{\ell}$ are the *frozen* unspent coins that will be used to claim stake value (hence voting power); S-Cond is the required data that satisfies all the stake attributes of $\{\mathsf{S}_i\}_{i=1}^{\ell}$; vk is a freshly generated signature verification key; and Addr is the return address for the voter to receive treasury reward. The voter's ID denoted as $\mathsf{V}_i := \mathsf{hash}(\mathsf{vk})$ is defined as the hash of vk.

Note that, the registration transaction is valid if and only if the payment conditions of all the input coins are satisfied. Thus, this indicates that a registered voter may represent a group of users' if the deposited coins in the registration transaction come from different users. Moreover, the return address is especially useful and distinct for a set of stakeholders that pool resources together to participate in the treasury system. This allows for flexibility in payment of participation reward to an address agreed to by all parties rather than to a fixed address. The address can have protective features such as multi-signature spend requirements to prevent all parties in the pool e.g. Voting Service Providers in Decred Treasury [Decc].

Let $\beta_{\min}$ be a predefined treasury system parameter. To register as an expert, a stakeholder (or a set of stakeholders) need(s) to deposit *exact* $\beta_{\min}$ amount of coins, by submitting a special *expert registration transaction* in the following form:

$$\mathsf{Tx}\Big(\{\mathsf{In}_i\}_{i=1}^{n}; \mathsf{TCoin}; \{\textsc{Expert-Reg}, \mathsf{TID}, \mathsf{vk}, \mathsf{Addr}\}\Big) \ ,$$

where $\{\mathsf{In}_i\}_{i=1}^{n}$ are the input coins, and TCoin is a special output coin whose spending condition is defined in Section 3.5.5. Moreover, for the coin value, the following condition must hold:

$$\mathsf{TCoin.Value} \geq \beta_{\min} \ .$$

In the $\mathsf{Payload}$ field, Expert-Reg is a tag that indicates it is a special transaction for expert registration; TID is the treasury ID that is used to uniquely identify a treasury period; vk is a freshly generated signature key; and Addr is the return address for the expert to receive treasury reward. The expert's ID denoted as $\mathsf{E}_j := \mathsf{hash}(\mathsf{vk})$ is defined as the hash of vk.

Note that the expert does not gain reward based on the amount of deposited coins, so it is not rational to deposit significantly more than $\beta_{\min}$ coins in practice. Specifically, expert rewards are based on the amount of received delegated votes.

3.5.4 Voting Committee Selection

At the beginning of the voting epoch, the voting committee of the previous treasury epoch jointly reveal the committed seed. The seed serves as an unbiased source of randomness for the cryptographic operation of selecting the new committee members (from all eligible participants).

Let $\mathsf{st}_i = \sum_{j=1}^{\ell} S_j.\mathsf{Value}$ for all the stake coins S_j claimed in the payload of a voter registration transaction of vk_i, i.e. st_i is the total stake amount claimed by vk_i. Once seed is announced, any registered voter, with address vk_i and claimed stake st_i, can volunteer

to participate in the voting committee if the following inequality is satisfied:

$$\mathsf{hash}\big(\mathsf{vk}_i, \mathsf{sign}_{\mathsf{sk}'_i}(\mathsf{seed})\big) \leq \mathsf{st}_i \cdot T$$

where sk'_i is the corresponding signing key for vk_i, and T is a pre-defined threshold for the blockchain treasury system. The threshold T is properly defined to ensure that approximately $\lambda' = \omega(\log \lambda)$, for example, $\lambda' = \mathrm{polylog}(\lambda)$ committee members are selected, assuming a constant fraction of them will be active. Note that, as is the case with most *proof-of-stake* systems, T needs to be updated frequently to ensure required security and efficiency performance. Chepurnoy et. al., in [CDFZ17] provides a good example of a method for threshold/difficulty T adjustment.

When the above inequality holds, the voter can then submit a special *registration transaction* in the following form:

$$\mathsf{Tx}\Big(\{\mathsf{In}_i\}_{i=1}^n; \mathsf{TCoin}; \Big\{\text{VC-REG}, \mathsf{TID}, \overline{\mathsf{vk}}, \tilde{\mathsf{pk}}, \mathsf{sign}_{\mathsf{sk}'_i}(\mathsf{seed}), \mathsf{Addr}\Big\}\Big) \ ,$$

where $\{\mathsf{In}_i\}_{i=1}^n$ are the input coins, and TCoin is a special output coin whose spending condition is defined in Section 3.5.5. Moreover, for the coin value, the following condition must hold:

$$\mathsf{TCoin.Value} \geq \gamma_{\min} \ .$$

In the $\mathsf{Payload}$ field, VC-REG is a tag that indicates it is a special voting committee registration transaction; TID is the treasury ID used to uniquely identify a treasury period; $\overline{\mathsf{vk}}$ is a freshly generated signature verification key; $\tilde{\mathsf{pk}}$ is a freshly generated public key for a pre-defined public key cryptosystem (lifted ElGamal); $\mathsf{sign}_{\mathsf{sk}'_i}(\mathsf{seed})$ is the signature of seed under the signing key corresponding to vk_i; and Addr is the return address for the committee member to receive treasury reward.

We remark that, honest majority of the voting committee is required to guarantee voter privacy and protocol termination. Assuming the majority of the stake of all the registered voters is honest, therefore, the probability that a selected committee member

is honest is

$$p = 1/2 + \varepsilon, \quad \text{for any} \quad \varepsilon \in (0, 1/2].$$

Let X be the number of malicious committee members selected among all λ' committee members. Since $\lambda' = \omega(\log \lambda)$, based on Chernoff bound, we have that, for $\delta = 2\varepsilon/(1-2\varepsilon)$:

$$\begin{aligned} \Pr[X \geq \lambda'/2] &= \Pr[X \geq (1+\delta)(1/2-\varepsilon)\lambda'] \\ &< \exp(-\delta^2(1/2-\varepsilon)\lambda'/4) \\ &= \frac{1}{\exp(\omega(\log \lambda))} = \mathsf{negl}(\lambda) \end{aligned}$$

Thus, the Chernoff bound allows us to control the probability of a malicious majority of committee members.

3.5.5 Supplying the Treasury

As earlier discussed in Section 2.6, treasury funding is perhaps the most crucial ingredient in a decentralised community-controlled decision-making system. Treasury funding should be secure, regular, sustainable and centralisation-free. That is, source of funding for treasury system should not introduce centralisation into the system.

Note that, although, individually not all the potential funding sources possess the properties mentioned above, a clever combination of some (or all) of these sources satisfy the set out requirements listed above. Therefore, we propose 3 main sources of funding for the treasury system.

- *Taxation/Haircut from block reward:* Most blockchain platforms offer block rewards (including transaction fees) to proposers of new blocks, incentivizing honest behaviour. A fraction of such block rewards can be taken and contributed to the decentralised treasury. This type of funding source is sustainable as long as the block rewards of the underlying blockchain platform remain. However, block rewards may fluctuate (reduce) over time, and it could cause a reduction in the amount of funds available to the treasury.

- *Minting new coins:* Coin minting represents, perhaps, the most sustainable funding source of the potential sources. At the beginning of each treasury period, certain amount of new coins are created to fund projects. However, minting must be properly analysed prior to adoption because it may cause inflation in terms of the fiat market value of the underlying cryptocurrency.

- *Donations or charity:* Treasury funds can also be supplemented by donations although given the ad-hoc nature of donations they are unsuitable and unsustainable as the main (or only) source funding cryptocurrency treasury system.

Treasury funds are accumulated via a collection of coins. For instance, during each treasury period, taxation of block rewards, minting and donations can be collected through a special transaction at the beginning of each block. This special transaction outputs new coins whose spending condition, Cond, specifies that the coins can only be spent according to the corresponding treasury decision. Treasury funds will be distributed in forms of transactions jointly made by the corresponding voting committee. Therefore, the accumulated coins for a given treasury period must allow the voting committee in that treasury period to jointly spend them.

Formally, there are λ' committee members selected at the beginning of the voting epoch of each treasury period. Let $\mathsf{seed}_{\mathsf{TID}_i}$ denote the seed opened in the treasury period indexed by TID_i. Let $\{\overline{\mathsf{vk}}_j\}_{j=1}^{\ell}$ be the set of signature verification keys in the valid committee registration transactions proposed by vk_i such that the following condition holds:

$$\mathsf{hash}\big(\mathsf{vk}_i, \mathsf{sign}_{\mathsf{sk}'_i}(\mathsf{seed})\big) \leq \mathsf{st}_i \cdot T$$

Therefore, the treasury coin spending condition is specified such that, a treasury coin can only be spent in a transaction if the majority of the signatures with respect to $\{\overline{\mathsf{vk}}_j\}_{j=1}^{\ell}$ are present in the transaction.

3.5.6 Enabling Stake Delegation

In our treasury system, we distinguish between the stake ownership and coin ownership. In other words, the stake of a coin can be 'owned' by a user other than the owner of the coin. This distinction enables the delegation of the stake of a coin to a party (other than the owner) without transferring the ownership of the coin. To realise this important feature, we introduce a stake attribute, denoted as S-Attr, that can be attached to the Payload of a coin. Any user/party who can provide the required data that satisfies the condition(s) in the S-Attr is able to claim the stake (not own) of the coin.

Of course, the stake of an unspent coin can only be claimed at most once at any moment (treasury period). In practice, to ensure secure realisation additional checks might be executed. If a user A wants to delegate the stake of a coin to a user B, he simply needs to put the user B's desired S-Attr in the Payload of the coin. Note that this type of delegation is persistent in the sense that if the coin is not consumed, the S-Attr of the coin remains the same. This feature allows users to stay offline while their coins' stake can still be used in the treasury process by the delegatees. Hence, encouraging increased stake (user) participation within treasury system decision-making. Recall that voting power within the treasury system is proportional to the amount of locked stake. However, this type of delegation only guarantees pseudonymity-based privacy, because anyone can learn who the owner of the coin stake is, by checking the S-Attr of the coin.

3.5.7 Handling the Treasury Specific Data in the Payload

Typically, validation rules for most general blockchain transactions do not take into account the content of transaction payload. Therefore, for treasury system transactions with useful payload data, additional checks are required to support these treasury special transactions. Specifically, we verify the payload data of those treasury system transactions with additional algorithms.

For instance, a coin must be *frozen* during an entire treasury period in order to claim its stake in the voting process. For example, this can be done by, adding extra constraints in the spending condition, ensuring that the coin cannot be spent until a given block

height, which is no earlier than the end of the treasury period. Furthermore, the stake of one coin can only be claimed once during each treasury period to guarantee users' voting power is tied to real (unused) stake and prevent Sybil attacks.

3.5.8 Decision-making

The voting committee members, voters, and experts follow the protocol described in Section 3.6 for the decision-making process. The described protocol covers key generation stage, the ballot casting stage, and the tally stage. In terms of security, as shown in Section 3.5.4, with overwhelming probability, the majority of the committee members are honest. Hence, guaranteeing voter privacy and protocol termination. In an unlikely extreme case, where all voting committee members are corrupt, our voting scheme still ensures the integrity of the voting tally result through the non-interactive zero-knowledge proofs which can be publicly verified by any party. As a further deterrent to prevent malicious behaviour from voting committee members, if a cheating voting committee member is detected, their locked stake deposited is seized (not returned to them).

Voters/experts need to submit an independent ballot for each project proposal. The voter's ballot is either a delegation of his voting power to an expert or a direct expression of his choice/decision on the project. However, an expert's ballot is a direct vote on the project. In our prototype treasury system implementation, we adopt the 'YES-NO-ABSTAIN' voting scheme. Specifically, after the voting by all eligible parties (voters and experts), the project proposals are scored based on the number of yes votes minus the number of no votes.

For a project to be shortlisted for potential funding from the treasury system, its net vote received must be least 10% of all votes in treasury period. Proposals that do not meet the net vote threshold are discarded. Thereafter, shortlisted proposals are ranked according to their score, and the top ranked proposals are funded in turns until available treasury fund is expended. Each of the voting committee members will then sign the treasury decision and transactions (payment to winning projects), and those transactions are valid if and only if they are signed by more than t-out-of-k voting committee members.

3.5.9 Post-voting Execution

Certain proportion (e.g. 20%) of the treasury fund will be used to reward the voting committee members, voters and experts for participation in the treasury system. A voter $\mathsf{V}_i \in \mathcal{V}$ will receive reward that is proportional to his/her locked stake, denoted as $\zeta_2 \cdot \mathsf{st}_i$, where st_i is the amount of the stake claimed by V_i. The expert $\mathsf{E}_j \in \mathcal{E}$ will receive reward that is proportional to his/her received delegations, denoted as $\zeta_3 \cdot D_j$, where D_j is the amount of delegations that E_j received. The voting committee members $\mathsf{C}_\ell \in \mathcal{C}$ will receive a fixed amount of reward, denoted as ζ_1. Note that as the voting committee members are required to perform additional actions in the next treasury period (seed revelation), their reward will only be transferred after the completion of those actions at the end of pre-voting epoch in the next treasury period. Meanwhile, if a voting committee member cheats or an expert fails to submit a valid ballot, he/she will lose their deposited coin as a penalty.

Specifically for the additional tasks carried out by voting committee members, they will jointly generate and commit to a random seed for the next treasury period, in a protocol depicted as follows. To generate and commit a random seed seed, voting committee members C_ℓ, $\ell \in [k]$ need to invoke a coin flipping protocol. However, given that the committee members already jointly setup a public key ap, the cost of such a protocol is very small. Specifically, in the final epoch of each treasury period, each voting committee member C_ℓ, $\ell \in [k]$ will pick a random group element $R_\ell \leftarrow \mathbb{G}$ and posts its encryption, $C_\ell \leftarrow \mathsf{Enc}_{\mathsf{pk}}(R_\ell)$ to the blockchain. $C := \prod_{\ell=1}^{k} C_\ell$ is defined as the committed/encrypted seed for the next treasury period. Note that C can then be jointly decrypted as far as majority of the voting committee members are honest (threshold decryption). This two stage-process for seed generation helps guarantee that malicious voting committee members cannot influence the distribution of the seed.

3.5.10 Partitionary Budgeting

As earlier established, the main goal of the treasury system is decentralized community-driven self-sustainable cryptocurrency development through adoption and funding of

projects. Naively, selection of projects for funding is done by ranking all submitted proposals according to the number of votes they receive. Subsequently, fund a number of projects whose total funding request does not exceed the available treasury budget. However, in practice, there exists a risk of underfunding vital areas of cryptocurrency development due to numerous project submissions and inflated discussions and interests in some other areas. To prevent this undesirable scenario, project proposals are classified into different categories and each category is independently allocated a percentage of available treasury funds. Thus, guaranteeing funds to all vital areas of cryptocurrency development (as long as voting threshold is satisfied by projects requesting treasury funds).

Analysis of existing blockchain development funding [KN$^+$,Fou,Decb] reveal marketing, PR, integration, software development and organisational costs are most prominent categories. Therefore, given this background and general business development rules, we propose (at least) the following categories.

- **Marketing :** This covers activities devoted to cryptocurrency market share growth; market analysis, advertisement, conferences, etc. The vastness of the area demands this category should take a sizeable percentage of the available funding budget.
- **Technology adoption :** This includes costs needed for wider spreading of cryptocurrency; integration with various platforms, websites and applications, deployment of ATMs, etc.
- **Development and security :** This includes costs allocated for funding core and non-core development, security incident response, patch management, running testnets, as well as similar critical technology areas.
- **Support :** This category includes user support, documentation, maintaining of web-infrastructure needed for the community and other similar areas.
- **Organization and management :** This category includes costs on team coordination and management, legal support, etc.

- **General :** This includes projects not covered by the above-listed categories, e.g. research on prospective technologies for cryptocurrency application, external security audit, collaboration with other communities, and charity.

It should be noted that the provided categories are recommendations and not an exhaustive list. Specifically, each cryptocurrency treasury deployment will factor in the peculiarities of its system, development stage, community, uses, law and regulations, in determining its applicable categories. For instance, a cryptocurrency in its early development stage, will assign greater preference towards software development, adoption and marketing efforts.

Nonetheless, having a categorisation approach to treasury system guarantees that critical areas for cryptocurrency routine operation, support and development will always receive funding via treasury, which in turn, guarantees cryptocurrency self-sustainability.

3.6 The proposed voting scheme

In this section, we formally present the decentralized voting scheme used in the treasury system. The voting scheme supports direct voting and vote delegation and the description done in the UC framework. In the following, the security model of the voting scheme is presented in Section 3.6.1.

3.6.1 Security Modeling

The entities involved in the voting schemes are a set of voting committee members $\mathcal{C} := \{\mathsf{C}_1, \ldots, \mathsf{C}_k\}$, a set of voters $\mathcal{V} := \{\mathsf{V}_1, \ldots, \mathsf{V}_n\}$, and a set of experts $\mathcal{E} := \{\mathsf{E}_1, \ldots, \mathsf{E}_m\}$. The security of the treasury voting scheme is analysed in the Universally Composable framework with static corruption. The security of the scheme is based on the indistinguishability between real/hybrid world executions and ideal world executions, i.e. for any PPT real/hybrid world adversary $\mathcal{A}$ we will construct an ideal world PPT simulator $\mathcal{S}$ that can present an indistinguishable view to the environment $\mathcal{Z}$ operating the protocol.

3.6.2 The Ideal World Execution

In the ideal world, the voting committee $\mathcal{C}$, the voters $\mathcal{V}$, and the experts $\mathcal{E}$ only communicate to an ideal functionality $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ during the execution. The ideal functionality $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ accepts a number of commands from $\mathcal{C}, \mathcal{V}, \mathcal{E}$. At the same time it informs the adversary of certain actions that take place and is also influenced by the adversary to elicit certain actions. The ideal functionality $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ is depicted in Fig. 3.10 and it consists of three phases: Preparation, Voting/Delegation, and Tally summarised as follows:

- **Preparation phase:** During the preparation phase, the voting committees $\mathsf{C}_i \in \mathcal{C}$ need to initiate the voting process by sending (INIT, sid) to the ideal functionality $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$. The voting will not start until all the committees have participated the preparation phase.

- **Voting/Delegation phase:** During the voting/delegation phase, the expert $\mathsf{E}_i \in \mathcal{E}$ can vote for his choice v_i by sending (VOTE, sid, v_i) to the ideal functionality $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$. Note that the voting choice v_i is leaked only when majority of the voting committees are corrupted. The voter $\mathsf{V}_j \in \mathcal{V}$, who owns α_j stake, can either vote directly for his choice v_j or delegate his voting power to an expert $\mathsf{E}_i \in \mathcal{E}$. Similarly, when all the voting committees are corrupted, $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ leaks the voters' ballots to the adversary $\mathcal{S}$.

- **Tally phase:** During tally phase, the voting committee $\mathsf{C}_i \in \mathcal{C}$ sends $(\text{DELCAL}, \mathsf{sid})$ to the ideal functionality $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ to calculate and reveal the delegations received by each expert. After that, they then send (TALLY, sid) to the ideal functionality $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ to open the tally. Once all the committees have opened the tally, any party can read the tally by sending $(\text{READTALLY}, sid)$ to $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$. Note that due to the nature of threshold cryptography, the adversary $\mathcal{S}$ can see the voting tally result before all the honest parties. Hence, the adversary can refuse to open the tally depending on the tally result. The tally algorithm $\mathsf{TallyAlg}$ is described in Fig. 3.13.

The ideal functionality $\mathcal{F}_{\mathbf{Vote}}^{t,k,n,m}$

The functionality $\mathcal{F}_{\text{VOTE}}^{t,k,n,m}$ interacts with a set of voting committees $\mathcal{C} := \{\mathsf{C}_1, \ldots, \mathsf{C}_k\}$, a set of voters $\mathcal{V} := \{\mathsf{V}_1, \ldots, \mathsf{V}_n\}$, a set of experts $\mathcal{E} := \{\mathsf{E}_1, \ldots, \mathsf{E}_m\}$, and the adversary $\mathcal{S}$. It is parameterized by a delegation calculation algorithm DelCal (described in Fig. 3.12) and a tally algorithm $\mathsf{TallyAlg}$ (described in Fig. 3.13) and variables ϕ_1, ϕ_2, τ, J_1, J_2, J_3, T_1 and T_2. Denote $\mathcal{C}_{\mathsf{cor}}$ and $\mathcal{C}_{\mathsf{honest}}$ as the set of corrupted and honest voting committees, respectively.

Initially, $\phi_1 = \emptyset$, $\phi_2 = \emptyset$, $\tau = \emptyset$, $J_1 = \emptyset$, $J_2 = \emptyset$, and $J_3 = \emptyset$.

Preparation:

- Upon receiving $(\text{INIT}, \mathsf{sid})$ from the voting committee $\mathsf{C}_i \in \mathcal{C}$, set $J_1 := J_1 \cup \{\mathsf{C}_i\}$, and send a notification message $(\text{INITNOTIFY}, \mathsf{sid}, \mathsf{C}_i)$ to the adversary $\mathcal{S}$.

Voting/Delegation:

- Upon receiving $(\text{VOTE}, \mathsf{sid}, v_i)$ from the expert $\mathsf{E}_i \in \mathcal{E}$, if $|J_1| < t$, ignore the request. Otherwise, record $(\mathsf{E}_i, \text{VOTE}, v_i)$ in ϕ_1; send a notification message $(\text{VOTENOTIFY}, \mathsf{sid}, \mathsf{E}_i)$ to the adversary $\mathcal{S}$. If $|\mathcal{C}_{\mathsf{cor}}| \geq t$, then additionally send a message $(\text{LEAK}, \mathsf{sid}, \mathsf{E}_i, \text{VOTE}, v_i)$ to the adversary $\mathcal{S}$.

- Upon receiving $(\text{CAST}, \mathsf{sid}, v_j, \alpha_j)$ from the voter $\mathsf{V}_j \in \mathcal{V}$, if $|J_1| < t$, ignore the request. Otherwise, record $(\mathsf{V}_j, \text{CAST}, v_j, \alpha_j)$ in ϕ_2; send a notification message $(\text{CASTNOTIFY}, \mathsf{sid}, \mathsf{V}_j, \alpha_j)$ to the adversary $\mathcal{S}$. If $|\mathcal{C}_{\mathsf{cor}}| \geq t$, then additionally send a message $(\text{LEAK}, \mathsf{sid}, \mathsf{V}_j, \text{CAST}, v_j)$ to the adversary $\mathcal{S}$.

Tally:

- Upon receiving $(\text{DELCAL}, \mathsf{sid})$ from the voting committee $\mathsf{C}_i \in \mathcal{C}$, set $J_2 := J_2 \cup \{\mathsf{C}_i\}$, and send a notification message $(\text{DELCALNOTIFY}, \mathsf{sid}, \mathsf{C}_i)$ to the adversary $\mathcal{S}$.

- If $|J_2 \cup \mathcal{C}_{\mathsf{honest}}| + |\mathcal{C}_{\mathsf{cor}}| \geq t$, send $(\textsc{LeakDel}, \mathsf{sid}, \mathsf{DelCal}(\mathcal{E}, \phi_2))$ to $\mathcal{S}$.
- If $|J_2| \geq t$, set $\delta \leftarrow \mathsf{DelCal}(\mathcal{E}, \phi_2)$.
- Upon receiving $(\textsc{Tally}, \mathsf{sid})$ from the voting committee $\mathsf{C}_i \in \mathcal{C}$, set $J_3 := J_3 \cup \{\mathsf{C}_i\}$, and send a notification message $(\textsc{TallyNotify}, \mathsf{sid}, \mathsf{C}_i)$ to the adversary $\mathcal{S}$.
- If $|J_3 \cup \mathcal{C}_{\mathsf{honest}}| + |\mathcal{C}_{\mathsf{cor}}| \geq t$, send $(\textsc{LeakTally}, \mathsf{sid}, \mathsf{TallyAlg}(\mathcal{V}, \mathcal{E}, \phi_1, \phi_2, \delta))$ to $\mathcal{S}$.
- If $|J_3| \geq t$, set $\tau \leftarrow \mathsf{TallyAlg}(\mathcal{V}, \mathcal{E}, \phi_1, \phi_2, \delta)$.
- Upon receiving $(\textsc{ReadTally}, \mathsf{sid})$ from any party, if $\delta = \emptyset \wedge \tau = \emptyset$ ignore the request. Otherwise, return $(\textsc{ReadTallyReturn}, \mathsf{sid}, (\delta, \tau))$ to the requester.

Figure 3.10: The ideal functionality $\mathcal{F}_{\text{VOTE}}^{t,k,n,m}$

3.6.3 The Real/Hybrid World Execution

In the real/hybrid world, the treasury voting scheme utilises a number of supporting components modelled as ideal functionalities. Specifically, the underlying blockchain infrastructure upon which the treasury system is built is modeled as ideal blockchain functionality [BMTZ17]. Moreover, the key generation functionality $\mathcal{F}_{\text{DKG}}^{t,k}$ [Wik04] is adopted for threshold key generation of the underlying public key cryptosystem. Finally, a global clock functionality $\mathcal{G}_{\text{CLOCK}}$ [BMTZ17] is adopted to model the synchronised network environment.

Let $\mathsf{EXEC}_{\Pi,\mathcal{A},\mathcal{Z}}$ denote the output of the environment $\mathcal{Z}$ when interacting with parties running the protocol Π and real-world adversary $\mathcal{A}$. Let $\mathsf{EXEC}_{\mathcal{F},\mathcal{S},\mathcal{Z}}$ denote the output of $\mathcal{Z}$ when running protocol ϕ interacting with the ideal functionality $\mathcal{F}$ and the ideal adversary $\mathcal{S}$.

Definition 7. *We say that a protocol* Π *UC-realizes an ideal functionality* $\mathcal{F}$ *if for any*

real/hybrid world adversary $\mathcal{A}$ there exists an ideal world adversary $\mathcal{S}$ such that for any environment $\mathcal{Z}$ that follows the rules of interaction for Universally Composable security we have:

$$\mathsf{EXEC}_{\Pi,\mathcal{A},\mathcal{Z}} \approx \mathsf{EXEC}_{\mathcal{F},\mathcal{S},\mathcal{Z}} \ .$$

3.6.4 The Voting Scheme

Let m be the number of experts and n be the number of voters. Let $\mathbf{e}_i^{(m)} \in \{0,1\}^m$ be the unit vector whose i-th coordinate is 1 and the remaining coordinates are 0. Furthermore, we also use the notation $\mathbf{e}_0^{(\ell)}$ to denote an ℓ-vector that contains 0 in all its coordinates. We use $\mathsf{Enc}_{\mathsf{pk}}(\mathbf{e}_i^{(\ell)})$ to denote coordinate-wise encryption of $\mathbf{e}_i^{(\ell)}$, i.e.

$$\mathsf{Enc}_{\mathsf{pk}}(\mathbf{e}_i^{(\ell)}) := \mathsf{Enc}_{\mathsf{pk}}(e_{i,1}^{(\ell)}), \ldots, \mathsf{Enc}_{\mathsf{pk}}(e_{i,\ell}^{(\ell)}),$$

where $\mathbf{e}_i^{(\ell)} = (e_{i,1}^{(\ell)}, \ldots, e_{i,\ell}^{(\ell)})$.

3.6.4.1 Vote Encoding

In the voting scheme, every vote is encoded into a unit vector. When a vote is from an expert, it is encoded as the unit vector 100 when expert's choice is Yes, 010 for No and 001 when the input from the expert is Abstain. However, for other voters, the vote is encoded as a unit-vector of length m+3, where m is the number of experts in the system. The vote is encoded as a unit vector with 1 in the coordinate i, for $i \in [m]$ where a voter delegates their vote. Similarly, where a voter votes directly by themselves, the vote is encoded as a unit vector with 1 in the coordinate i for $i \in [m+1, \ldots, m+3]$.

Formally, let $\mathsf{encode}^{\mathsf{E}}$ and $\mathsf{encode}^{\mathsf{V}}$ be the vote encoding algorithm for the expert and voter, respectively. For an expert, upon receiving input $x \in \{\textsc{Yes}, \textsc{No}, \textsc{Abstain}\}$, the $\mathsf{encode}^{\mathsf{E}}$ returns 100, 010, 001 for $\textsc{Yes}, \textsc{No}, \textsc{Abstain}$, respectively. However, for a voter, the input is $y \in \{\mathsf{E}_1, \ldots, \mathsf{E}_m\} \cup \{\textsc{Yes}, \textsc{No}, \textsc{Abstain}\}$. When $y = \mathsf{E}_i$, for $i \in [m]$, it implies that the voter delegates his/her voting power to the expert E_i i.e. expert whose index is i. Alternatively, when $y \in \{\textsc{Yes}, \textsc{No}, \textsc{Abstain}\}$, it implies that the voter directly votes on the project.

The $\mathsf{encode}^{\mathsf{V}}$ returns a unit vector of length $(m+3)$, denoted as v, such that $v = e_i^{(m+3)}$ if $y = \mathsf{E}_i$, for $i \in [m]$; and v is set to $e_{m+1}^{(m+3)}$, $e_{m+2}^{(m+3)}$, and $e_{m+3}^{(m+3)}$ if y is YES, NO, ABSTAIN, respectively.

Note that since sending data to the blockchain consumes coins, it is implicitly assumed that all the experts $\mathcal{E}$ and voters $\mathcal{V}$ have spare coins to pay the transaction fees that is incurred during the protocol execution. Given that participation in the voting scheme is incentivised, voters and experts can reclaim the costs in the form of treasury reward. Specifically, we let each party prepare $\{\mathsf{In}_i\}_{i=1}^{\ell_1}, \{\mathsf{Out}_j\}_{j=1}^{\ell_2}$ such that:

$$\sum_{i=1}^{\ell_1} \mathsf{In}_i.\mathsf{Value} \geq \sum_{j=1}^{\ell_2} \mathsf{Out}_j.\mathsf{Value} \ .$$

The corresponding coins owned by a voter $\mathsf{V}_i \in \mathcal{V}$, an expert $\mathsf{E}_j \in \mathcal{E}$, and a voting committee member $\mathsf{C}_t \in \mathcal{C}$ is denoted as $(\{\mathsf{In}_\eta^{(\mathsf{V}_i)}\}_{\eta=1}^{\ell_1}, \{\mathsf{Out}_\eta^{(\mathsf{V}_i)}\}_{\eta=1}^{\ell_2})$, $(\{\mathsf{In}_\eta^{(\mathsf{E}_j)}\}_{\eta=1}^{\ell_1}, \{\mathsf{Out}_\eta^{(\mathsf{E}_j)}\}_{\eta=1}^{\ell_2})$, and $(\{\mathsf{In}_\eta^{(\mathsf{C}_t)}\}_{\eta=1}^{\ell_1}, \{\mathsf{Out}_\eta^{(\mathsf{C}_t)}\}_{\eta=1}^{\ell_2})$, respectively.

The vote encoding scheme is used in the voting protocol depicted in Fig. 3.11 and is relevant to the voting/delegation phase, and tally phase of the voting protocol.

The voting protocol $\Pi_{\mathbf{Vote}}^{t,k,m,n}$

Denote the corresponding coins owned by a voter $\mathsf{V}_i \in \mathcal{V}$, an expert $\mathsf{E}_j \in \mathcal{E}$, and a voting committee member $\mathsf{C}_t \in \mathcal{C}$ as $(\{\mathsf{In}_\eta^{(\mathsf{V}_i)}\}_{\eta=1}^{\ell_1}, \{\mathsf{Out}_\eta^{(\mathsf{V}_i)}\}_{\eta=1}^{\ell_2})$, $(\{\mathsf{In}_\eta^{(\mathsf{E}_j)}\}_{\eta=1}^{\ell_1}, \{\mathsf{Out}_\eta^{(\mathsf{E}_j)}\}_{\eta=1}^{\ell_2})$, and $(\{\mathsf{In}_\eta^{(\mathsf{C}_t)}\}_{\eta=1}^{\ell_1}, \{\mathsf{Out}_\eta^{(\mathsf{C}_t)}\}_{\eta=1}^{\ell_2})$, respectively.

Preparation phase:

- Upon receiving $(\textsc{Init}, \mathsf{sid})$ from the environment $\mathcal{Z}$, the committee C_j, $j \in [k]$ sends $(\textsc{KeyGen}, \mathsf{sid})$ to $\mathcal{F}_{\text{DKG}}^{t,k}$ to generate pk.

Voting/Delegation phase:

- Upon receiving $(\textsc{Vote}, \mathsf{sid}, v_j)$ from the environment $\mathcal{Z}$, the expert E_j, $j \in [m]$ does the following:
  - Send $(\textsc{ReadPK}, \mathsf{sid})$ to $\mathcal{F}_{\text{DKG}}^{t,k}$, and receive $(\textsc{PublicKey}, \mathsf{sid}, \mathsf{pk})$ from $\mathcal{F}_{\text{DKG}}^{t,k}$.

- Set the unit vector $\mathbf{e}^{(3)} \leftarrow \mathsf{encode}^{\mathsf{E}}(v_j)$. Compute $\mathbf{c_j}^{(3)} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathbf{e}^{(3)})$ and its NIZK proof π_j (see Section 3.7).
- Execute macro $\mathsf{Send\text{-}Msg}\Big((\mathbf{c_j}^{(3)}, \pi_j), \{\mathsf{In}_\eta^{(\mathsf{E}_j)}\}_{\eta=1}^{\ell_1}, \{\mathsf{Out}_\eta^{(\mathsf{E}_j)}\}_{\eta=1}^{\ell_2}\Big)$. (see Fig. 3.14)

- Upon receiving $(\textsc{Cast}, \mathsf{sid}, v_i, \alpha_i)$ from the environment $\mathcal{Z}$, the voter V_i, $i \in [n]$ does the following:
  - Send $(\textsc{ReadPK}, \mathsf{sid})$ to $\mathcal{F}_{\text{DKG}}^{t,k}$, and receive $(\textsc{PublicKey}, \mathsf{sid}, \mathsf{pk})$ from $\mathcal{F}_{\text{DKG}}^{t,k}$.
  - Set the unit vector $\mathbf{e}^{(m+3)} \leftarrow \mathsf{encode}^{\mathsf{V}}(v_i)$. Compute $\mathbf{u_i}^{(m+3)} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathbf{e}^{(m+3)})$ and its NIZK proof σ_i (see Section 3.7).
  - Execute macro $\mathsf{Send\text{-}Msg}\Big((\mathbf{u_i}^{(m+3)}, \sigma_i, \alpha_i), \{\mathsf{In}_\eta^{(\mathsf{V}_i)}\}_{\eta=1}^{\ell_1}, \{\mathsf{Out}_\eta^{(\mathsf{V}_i)}\}_{\eta=1}^{\ell_2}\Big)$. (see Fig. 3.14)

Tally phase:

- Upon receiving $(\textsc{DelCal}, \mathsf{sid})$ from the environment $\mathcal{Z}$, the committee C_t, $t \in [k]$ does:
  - Execute macro $\mathsf{Read\text{-}Msg}$ and obtain data.
  - Fetch the ballots $\{(\mathbf{c_i}^{(3)}, \pi_i)\}_{i\in[m]}$ and $\{(\mathbf{u_j}^{(m+3)}, \sigma_j, \alpha_j)\}_{j\in[n]}$ from data.
  - For $i \in [m]$, check $\mathsf{Verify}(\mathbf{c_i}^{(3)}, \pi_i) = 1$; for $j \in [n]$, $\mathsf{Verify}(\mathbf{u_j}^{(m+3)}, \sigma_j) = 1$. Remove all the invalid ballots.
  - For $j \in [n]$, if a valid $\mathbf{u_j}^{(m+3)}$ is posted, parse $\mathbf{u_j}^{(m+3)}$ to $(\mathbf{a_j}^{(m)}, \mathbf{b_j}^{(3)})$.
  - For $j \in [n]$, $\ell \in [0, m-1]$, compute $z_{i,\ell} := a_{j,\ell}^{\alpha_j}$.
  - For $i \in [0, m-1]$, compute $s_i := \prod_{\ell=1}^{n} z_{\ell,i}$ and jointly decrypt it to w_i (see [GJKR99]).

- Upon receiving $(\textsc{Tally}, \mathsf{sid})$ from the environment $\mathcal{Z}$, the committee C_t, $t \in [k]$ does:

  - For $i \in [0, m-1]$, $\ell \in [0, 2]$, compute $d_{i,\ell} := c_{i,\ell}^{w_i}$.
  - For $\ell \in [0, 2]$, compute $x_\ell := \prod_{j=0}^{m-1} d_{j,\ell} \cdot \prod_{j=1}^{n} b_{j,\ell}^{\alpha_j}$ and jointly decrypt it to y_ℓ (see [GJKR99]). Execute macro $\mathsf{Send\text{-}Msg}\Big((x_\ell, y_\ell), \{\mathsf{In}_\eta^{(\mathsf{C}_t)}\}_{\eta=1}^{\ell_1}, \{\mathsf{Out}_\eta^{(\mathsf{C}_t)}\}_{\eta=1}^{\ell_2}\Big)$. (see Fig. 3.14)

- Upon receiving $(\textsc{ReadTally}, \mathsf{sid})$ from the environment $\mathcal{Z}$, the party P does the following:
  - Execute macro $\mathsf{Read\text{-}Msg}$ and obtain data.
  - Fetch $\{(x_i, y_i)\}_{i\in[0,2]}$ from data, and return $(\textsc{ReadTallyReturn}, \mathsf{sid}, (y_0, y_1, y_2))$ to the environment $\mathcal{Z}$.

Figure 3.11: The voting protocol $\Pi_{\text{VOTE}}^{t,k,m,n}$ in $\{\mathcal{F}_{\text{LEDGER}}, \mathcal{F}_{\text{DKG}}^{t,k}\}$-hybrid model

Algorithm DelCal

Input: a set of the expert labels $\mathcal{E}$, and a set of ballots ϕ_2
Output: the delegation result δ

Init:

- For $i \in [1, m]$, create and initiate $D_i = 0$.

Delegation interpretation:

- For each ballot $B \in \phi_2$: parse B in form of $(\mathsf{V}_j, \textsc{Cast}, v_j, \alpha_j)$; if $v_j = (\mathsf{Delegate}, \mathsf{E}_i)$ for some $\mathsf{E}_i \in \mathcal{E}$, then $D_i := D_i + \alpha_j$.

Output:

- Return $\delta := \{(\mathsf{E}_i, D_i)\}_{i\in[m]}$.

Figure 3.12: The delegation calculation algorithm DelCal

The tally algorithm $\mathsf{TallyAlg}$

Input: a set of the voters $\mathcal{V}$, a set of the experts $\mathcal{E}$, two sets of ballots ϕ_1, ϕ_2 and the delegation δ.
Output: the tally result τ

Init:

- Create and initiate $\tau_{\mathsf{yes}} = 0$, $\tau_{\mathsf{no}} = 0$ and $\tau_{\mathsf{abstain}} = 0$.
- Parse δ as $\{(\mathsf{E}_i, D_i)\}_{i\in[m]}$.

Tally Computation:

- For each ballot $B \in \phi_2$: parse B in form of $(\mathsf{V}_j, \textsc{Cast}, v_j, \alpha_j)$; if $v_j = (\mathsf{Vote}, a_j)$ for some $a_j \in \{\mathsf{yes}, \mathsf{no}, \mathsf{abstain}\}$, then $\tau_{a_j} := \tau_{a_j} + \alpha_j$.
- For each ballot $B \in \phi_1$: parse B in form of $(\mathsf{E}_i, \textsc{Vote}, b_i)$ for some $b_i \in \{\mathsf{yes}, \mathsf{no}, \mathsf{abstain}\}$, then $\tau_{b_i} := \tau_{b_i} + D_i$.

Output:

- Return $\tau := (\tau_{\mathsf{yes}}, \tau_{\mathsf{no}}, \tau_{\mathsf{abstain}})$.

Figure 3.13: The tally algorithm $\mathsf{TallyAlg}$

3.6.4.2 Sending/Reading Data to/from $\mathcal{F}_{\mathsf{Ledger}}$

For parties to send and read data from the blockchain, we provide a macro depicted in Fig. 3.14. The macro captures the three main pertinent types of delay for sending and reading blockchain data in the blockchain model of [BMTZ17]. First, we have a bounded network delay, and assume that all messages can be delivered within Δ_1 rounds, which is $2\Delta_1$ clock-ticks in [BMTZ17]. Subsequently, a desynchronised user can get up-to-date within $2\Delta_1$ rounds (i.e. $4\Delta_1$ clock-ticks) after registration. The second type of delay captures the fact that the adversary can hold a valid transaction up to certain blocks, but the adversary cannot permanently deny service to (or DoS) such a transaction. Specifically, this is modeled by the $\mathsf{ExtendPolicy}$ in $\mathcal{F}_{\textsc{Ledger}}$, where if a transaction is more than Δ_2 rounds (i.e. $2\Delta_2$ clock-ticks) old, and is still valid with respect to the current state, then it will be included into the state. Finally, the third delay is $\mathsf{windowsize}$. This captures that the adversary can set state-slackness of all the honest parties up to the $\mathsf{windowsize}$, which is consistent with the *common prefix* property in [GKL15]. Hence,

all the honest parties can have a common state of any block that has been proposed for more than windowsize. The windowsize is denoted as Δ_3 rounds (i.e. $2\Delta_3$ clock-ticks).

To send a message x to $\mathcal{F}_{\text{LEDGER}}$, we need to first check that a party has deregistered and desynchronized. If true, the party needs to first send (REGISTER, sid) to $\mathcal{F}_{\text{LEDGER}}$. It should be noted that a registered but desynchronised party can still send a transaction before it is fully updated as synchronised. Effectively, the message x is stored in the payload of a 'dummy' transaction whose input coins and output coins share the same owner (spending condition).

To read a message (stored in the payload of some transaction) from $\mathcal{F}_{\text{LEDGER}}$, analogously a deregistered party needs to first send (REGISTER, sid) to $\mathcal{F}_{\text{LEDGER}}$. After $4\delta_1$ clock-ticks, the party can become synchronised. In order to receive the latest message, the party needs to wait a maximum of $2(\Delta_2 + \Delta_3)$ clock-ticks for the transaction that carries the intended message to be included in the state of the party.

3.7 A Novel Unit Vector ZK Proof

We denote a unit vector of length n as: $\mathbf{e}_i^{(n)} = (e_{i,0}, \ldots, e_{i,n-1})$, where its i-th coordinate is 1 and the rest coordinates are 0. Conventionally, to show a vector of ElGamal ciphertexts element-wise encrypt a unit vector, Chaum-Pedersen proofs [CP93] are used to show each of the ciphertexts encrypts either 0 or 1 (via Sigma OR composition) and the product of all the ciphertexts encrypts 1. This type of proof is used in many well-known voting schemes, e.g. Helios [Adi08]. Although this proof works (is valid), the proof size is linear in the length of the unit vector. Thus the communication overhead is considerably significant when the unit vector length becomes larger.

In this section, we propose a novel special honest verifier zero-knowledge (SHVZK) proof for unit vector that allows the prover to convince the verifier that a vector of ciphertexts $(C_0, \ldots, C_{n-1})$ encrypts a unit vector $\mathbf{e}_i^{(n)}$, $i \in [0, n-1]$ with $O(\log n)$ proof size. Without loss of generality, assume n is a perfect power of 2. Otherwise, append $\mathsf{Enc}_{\mathsf{pk}}(0;0)$ (i.e. trivial ciphertexts) to take the total number of ciphertexts to the next

Sending and reading messages

Macro Send-Msg$(x, \{\mathsf{In}_i\}_{i=1}^{\ell_1}, \{\mathsf{Out}_j\}_{j=1}^{\ell_2})$**:**

- If the party has deregistered and desynchronized:
  - Send $(\textsc{Register}, \mathsf{sid})$ to $\mathcal{F}_{\textsc{Ledger}}$.
  - Send $\Big(\textsc{Submit}, sid, \mathsf{Tx}(\{\mathsf{In}_i\}_{i=1}^{\ell_1}; \{\mathsf{Out}_j\}_{j=1}^{\ell_2}; x)\Big)$ to $\mathcal{F}_{\textsc{Ledger}}$.
  - Send $(\textsc{De-Register}, \mathsf{sid})$ to $\mathcal{F}_{\textsc{Ledger}}$.
- If the party is already synchronized:
  - Send $\Big(\textsc{Submit}, sid, \mathsf{Tx}(\{\mathsf{In}_i\}_{i=1}^{\ell_1}; \{\mathsf{Out}_j\}_{j=1}^{\ell_2}; x)\Big)$ to $\mathcal{F}_{\textsc{Ledger}}$.

Macro Read-Msg**:**

- If the party has deregistered and desynchronized:
  - Send $(\textsc{Register}, \mathsf{sid})$ to $\mathcal{F}_{\textsc{Ledger}}$.
  - Wait for $\max\{4\Delta_1, 2(\Delta_2 + \Delta_3)\}$ clock-ticks by keeping sending $(\textsc{Tick}, \mathsf{sid})$ to the $\mathcal{G}_{\textsc{Clock}}$.
  - Send $(\textsc{Read}, sid)$ to $\mathcal{F}_{\textsc{Ledger}}$ and receive $(\textsc{Read}, sid, \mathsf{data})$ from $\mathcal{F}_{\textsc{Ledger}}$.
  - Send $(\textsc{De-Register}, \mathsf{sid})$ to $\mathcal{F}_{\textsc{Ledger}}$.
- If the party is already synchronized:
  - Wait for $\max\{4\Delta_1, 2(\Delta_2 + \Delta_3)\}$ clock-ticks by keeping sending $(\textsc{Tick}, \mathsf{sid})$ to the $\mathcal{G}_{\textsc{Clock}}$.
  - Send $(\textsc{Read}, sid)$ to $\mathcal{F}_{\textsc{Ledger}}$ and receive $(\textsc{Read}, sid, \mathsf{data})$ from $\mathcal{F}_{\textsc{Ledger}}$.
- Return data.

Figure 3.14: Macro for sending and receiving message via $\mathcal{F}_{\textsc{Ledger}}$

power of 2. Our proposed novel SHVZK protocol can also be Fiat-Shamir transformed to a non-interactive ZK (NIZK) proof in the random oracle model [BR93].

The basic idea of our construction is inspired by [GK15], where Groth and Kohlweiss proposed a Sigma protocol for the prover to show that he (the prover) knows how to open one out of many commitments. The key idea behind our construction is that there exists a data-oblivious algorithm that can take as input $i \in \{0,1\}^{\log n}$ and output the unit vector $\mathbf{e}_i^{(n)}$. Let $i_1, \ldots, i_{\log n}$ be the binary representation of i. The algorithm is depicted in Fig. 3.15.

The algorithm that maps $i \in [0, n-1]$ to $\mathbf{e}_i^{(n)}$

Input: index $i = (i_1, \ldots, i_{\log n}) \in \{0,1\}^{\log n}$
Output: unit vector $\mathbf{e}_i^{(n)} = (e_{i,0}, \ldots, e_{i,n-1}) \in \{0,1\}^n$

- For $\ell \in [\log n]$, set $b_{\ell,0} := 1 - i_\ell$ and $b_{\ell,1} := i_\ell$;
- For $j \in [0, n-1]$, set $e_{i,j} := \prod_{\ell=1}^{\log n} b_{\ell,j_\ell}$, where $j_1, \ldots, j_{\log n}$ is the binary representation of j;
- Return $\mathbf{e}_i^{(n)} = (e_{i,0}, \ldots, e_{i,n-1})$;

Figure 3.15: The algorithm that maps $i \in [0, n-1]$ to $\mathbf{e}_i^{(n)}$

Intuitively, we let the prover first bit-wisely commit the binary presentation of $i \in [0, n-1]$ for the unit vector $\mathbf{e}_i^{(n)}$. The prover then shows that each of the commitments of $(i_1, \ldots, i_{\log n})$ indeed contains a 0 or 1, using the Sigma protocol proposed in Section 2.3 of [GK15]. Note that in the 3rd move of such a Sigma protocol, the prover reveals a degree-1 polynomial of the committed message.

Let the corresponding degree-1 polynomials be denoted as:

$$z_{\ell,1} := i_\ell x + \beta_\ell \ , \quad \ell \in [\log n]$$

where β_ℓ are chosen by the prover and x is chosen by the verifier. By linearity, we can also define:

$$z_{\ell,0} := x - z_{\ell,1} = (1 - i_\ell)x - \beta_\ell, \qquad \ell \in [\log n].$$

According to the algorithm described in Fig.3.15, for $j \in [0, n-1]$, let $j_1, \ldots, j_{\log n}$ be the binary representation of j, and the product $\prod_{\ell=1}^{\log n} z_{\ell,j_\ell}$ can be viewed as a degree-$(\log n)$ polynomial of the form

$$p_j(x) = e_{i,j}x^{\log n} + \sum_{k=0}^{\log n-1} p_{j,k}x^k$$

for some $p_{j,k}$, $k \in [0, \log n - 1]$. We then use batch verification to show that each of C_j indeed encrypts $e_{i,j}$. Particularly, for a randomly chosen $y \leftarrow \mathbb{Z}_p$, let

$$E_j := (C_j)^{x^{\log n}} \cdot \mathsf{Enc}(-p_j(x); 0);$$

the prover needs to show that

$$E := \prod_{j=0}^{n-1}(E_j)^{y^j} \cdot \prod_{k=0}^{\log n-1}(D_k)^{x^k}$$

encrypts 0, where

$$D_\ell := \mathsf{Enc}_{\mathsf{pk}}(\sum_{j=0}^{n-1}(p_{j,\ell} \cdot y^j); R_\ell), \quad \ell \in [0, \log n - 1]$$

with fresh randomness $R_\ell \in \mathbb{Z}_p$.

Fig. 3.16 details our construction which consists of 5 moves. Both the prover and the verifier share a common reference string (CRS), which is a Pedersen commitment key that can be generated using random oracle. The prover first commits to each bits of the binary representation of i, and the commitments are denoted as I_ℓ, $\ell \in [\log n]$. Subsequently, it produces B_ℓ, A_ℓ as the first move of the Sigma protocol in Sec. 2.3 of [GK15] showing I_ℓ commits to 0 or 1. Jumping ahead, later the prover will receive a challenge $x \leftarrow \{0,1\}^\lambda$, and it then computes the third move of the Sigma protocol by producing

$$\{z_\ell, w_\ell, v_\ell\}_{\ell=1}^{\log n} .$$

To enable batch verification, before that, the prover is given another challenge $y \leftarrow \{0,1\}^\lambda$ in the second move. The prover computes and sends $\{D_\ell\}_{\ell=0}^{\log n-1}$. The verification

comprises two parts described as follows. In the first part, the verifier checks the following equations to ensure that I_ℓ commits to 0 or 1.

- $(I_\ell)^x \cdot B_\ell = \mathsf{Com}_{\mathsf{ck}}(z_\ell; w_\ell)$
- $(I_\ell)^{x-z_\ell} \cdot A_\ell = \mathsf{Com}_{\mathsf{ck}}(0; v_\ell)$

In the second part, the verifier checks if

$$\prod_{j=0}^{n-1} \left((C_j)^{x^{\log n}} \cdot \mathsf{Enc}_{\mathsf{pk}}(-\prod_{\ell=1}^{\log n} z_{\ell,j_\ell}; 0)\right)^{y^j} \cdot \prod_{\ell=0}^{\log n-1} (D_\ell)^{x^\ell}$$

is encryption of 0 by asking the prover to reveal the randomness.

Theorem 1. *The protocol described in Fig. 3.16 is a* 5*-move public coin special honest verifier zero-knowledge argument of knowledge of* $\mathbf{e}_i^{(n)} = (e_{i,0}, \ldots, e_{i,n-1}) \in \{0,1\}^n$ *and* $(r_0, \ldots, r_{n-1}) \in (\mathbb{Z}_p)^n$ *such that* $C_j = \mathsf{Enc}_{\mathsf{pk}}(e_{i,j}; r_j)$, $j \in [0, n-1]$ *under the DDH assumption.*

Proof. For perfect completeness, we first observe that the verification equations

$$(I_\ell)^x \cdot B_\ell = \mathsf{Com}_{\mathsf{ck}}(z_\ell; w_\ell)$$

and

$$(I_\ell)^{x-z_\ell} \cdot A_\ell = \mathsf{Com}_{\mathsf{ck}}(0; v_\ell)$$

holds. Indeed, by additively homomorphic property of the commitment scheme,

$$(I_\ell)^x \cdot B_\ell = \mathsf{Com}_{\mathsf{ck}}(i_\ell \cdot x + \beta_\ell; \alpha_\ell \cdot x + \gamma_\ell)$$

and

$$(I_\ell)^{x-z_\ell} \cdot A_\ell = \mathsf{Com}_{\mathsf{ck}}(i_\ell \cdot (x - z_\ell) + i_\ell \cdot \beta_\ell; \alpha_\ell \cdot (x - z_\ell) + \delta_\ell) = \mathsf{Com}_{\mathsf{ck}}(i_\ell(1 - i_\ell) \cdot x; v_\ell) \ .$$

Unit vector ZK argument

CRS: the commitment key ck
Statement: the public key pk and the ciphertexts $C_0 := \mathsf{Enc}_{\mathsf{pk}}(e_{i,0}; r_0), \ldots, C_{n-1} := \mathsf{Enc}_{\mathsf{pk}}(e_{i,n-1}; r_{n-1})$
Witness: the unit vector $\mathbf{e}_i^{(n)} \in \{0,1\}^n$ and the randomness $r_0, \ldots, r_{n-1} \in \mathbb{Z}_p$
Protocol:

- The prover P, for $\ell = 1, \ldots, \log n$, does:
  - Pick random $\alpha_\ell, \beta_\ell, \gamma_\ell, \delta_\ell \leftarrow \mathbb{Z}_p$;
  - Compute $I_\ell := \mathsf{Com}_{\mathsf{ck}}(i_\ell; \alpha_\ell)$, $B_\ell := \mathsf{Com}_{\mathsf{ck}}(\beta_\ell; \gamma_\ell)$ and $A_\ell := \mathsf{Com}_{\mathsf{ck}}(i_\ell \cdot \beta_\ell; \delta_\ell)$;
- $P \to V$: $\{I_\ell, B_\ell, A_\ell\}_{\ell=1}^{\log n}$;
- $V \to P$: Random $y \leftarrow \{0,1\}^\lambda$;
- The prover P for $\ell = 0, \ldots, \log n - 1$, does:
  - Pick random $R_\ell \leftarrow \mathbb{Z}_p$ and compute $D_\ell := \mathsf{Enc}_{\mathsf{pk}}\big(\sum_{j=0}^{n-1}(p_{j,\ell} \cdot y^j); R_\ell\big)$
- $P \to V$: $\{D_\ell\}_{\ell=0}^{\log n-1}$;
- $V \to P$: Random $x \leftarrow \{0,1\}^\lambda$;
- The prover P does the following:
  - Compute $R := \sum_{j=0}^{n-1}(r_j \cdot x^{\log n} \cdot y^j) + \sum_{\ell=0}^{\log n-1}(R_\ell \cdot x^\ell)$;
  - For $\ell = 1, \ldots, \log n$, compute $z_\ell := i_\ell \cdot x + \beta_\ell$, $w_\ell := \alpha_\ell \cdot x + \gamma_\ell$, and $v_\ell := \alpha_\ell(x - z_\ell) + \delta_\ell$;
- $P \to V$: R and $\{z_\ell, w_\ell, v_\ell\}_{\ell=1}^{\log n}$

Verification:

- Check the followings:
- For $\ell = 1, \ldots, \log n$, does:
  - $(I_\ell)^x \cdot B_\ell = \mathsf{Com}_{\mathsf{ck}}(z_\ell; w_\ell)$
  - $(I_\ell)^{x - z_\ell} \cdot A_\ell = \mathsf{Com}_{\mathsf{ck}}(0; v_\ell)$
- $\prod_{j=0}^{n-1}\left((C_j)^{x^{\log n}} \cdot \mathsf{Enc}_{\mathsf{pk}}(-\prod_{\ell=1}^{\log n} z_{\ell,j_\ell}; 0)\right)^{y^j} \cdot \prod_{\ell=0}^{\log n-1}(D_\ell)^{x^\ell} = \mathsf{Enc}_{\mathsf{pk}}(0; R)$, where $z_{j,1} = z_j$ and $z_{j,0} = x - z_j$.

Figure 3.16: Unit vector ZK argument

Since $i_\ell(1-i_\ell)=0$ when $i_\ell \in \{0,1\}$, we have

$$(I_\ell)^{x-z_\ell} \cdot A_\ell = \mathsf{Com}_{\mathsf{ck}}(0; v_\ell) \ .$$

Moreover, for each $j \in [0, n-1]$, $\prod_{\ell=1}^{\log n} z_{\ell,j_\ell}$ is a polynomial in the form of

$$p_j(x) = e_{i,j} x^{\log n} + \sum_{k=0}^{\log n - 1} p_{j,k} x^k$$

where x is the verifier's challenge. Therefore, it is easy to see

$$\begin{aligned}
&\prod_{j=0}^{n-1} \left((C_j)^{x^{\log n}} \cdot \mathsf{Enc}_{\mathsf{pk}}(-\prod_{\ell=1}^{\log n} z_{\ell,j_\ell}; 0)\right)^{y^j} \\
&\cdot \prod_{\ell=0}^{\log n-1} \mathsf{Enc}_{\mathsf{pk}}(\sum_{j=0}^{n-1}(p_{j,\ell} \cdot y^j); R_\ell)^{x^\ell} \\
&= \mathsf{Enc}_{\mathsf{pk}}\Big(\sum_{j=0}^{n-1} \big(e_{i,j} \cdot x^{\log n} - p_j(x) + \sum_{\ell=0}^{\log n-1} p_{j,\ell} \cdot x^\ell\big) \cdot y^j; R\Big) \\
&= \mathsf{Enc}_{\mathsf{pk}}(0; R) \ .
\end{aligned}$$

For soundness, first of all, the Sigma protocols for commitments of i_ℓ, $\ell \in [\log n]$ is specially sound, i.e., given two transactions with the same $\{I_\ell, B_\ell, A_\ell\}_{\ell=1}^{\log n}$ and two different x and $\{z_\ell, w_\ell, v_\ell\}_{\ell=1}^{\log n}$, there exists a PPT extractor that can output the corresponding witness $i_\ell \in \{0,1\}$.

Moreover,

$$\prod_{j=0}^{n-1} \left((C_j)^{x^{\log n}} \cdot \mathsf{Enc}_{\mathsf{pk}}(-\prod_{\ell=1}^{\log n} z_{\ell,j_\ell}; 0)\right)^{y^j}$$

builds a degree-$\log n$ polynomial with respect to x in the plaintext, while

$$\prod_{\ell=0}^{\log n-1} (D_\ell)^{x^\ell}$$

encrypts a degree-$(\log n - 1)$ polynomial w.r.t. x. Since x is randomly sampled after D_ℓ

is committed, Schwartz-Zippel lemma,

$$\prod_{j=0}^{n-1} \left((C_j)^{x^{\log n}} \cdot \mathsf{Enc}_{\mathsf{pk}}(-\prod_{\ell=1}^{\log n} z_{\ell,j_\ell}; 0)\right)^{y^j} \cdot \prod_{\ell=0}^{\log n-1} (D_\ell)^{x^\ell}$$

encrypts a zero polynomial with respect to x with overwhelming probability if the polynomial evaluation is 0. Therefore,

$$Q(y) := \sum_{j=0}^{n-1} (e_{i,j} - \prod_{\ell=1}^{\log n} i_{\ell,j_\ell}) \cdot y^j = 0$$

with overwhelming probability. Similarly, by Schwartz-Zippel lemma, $Q(y)$ is a zero polynomial. Hence, we have for $j \in [0, n-1]$,

$$e_{i,j} = \prod_{\ell=1}^{\log n} i_{\ell,j_\ell}$$

with overwhelming probability.

In terms of special honest verifier zero-knowledge (SHVZK), we now construct a simulator Sim that takes as input the statement $(C_0, \ldots, C_{n-1})$ and the given challenges $x, y \in \{0,1\}^\lambda$, and outputs a simulated transcript whose distribution is indistinguishable from the real one. Specifically, Sim first randomly picks $i_\ell \leftarrow \{0,1\}$ and

$$\alpha_\ell, \beta_\ell, \gamma_\ell, \delta_\ell \leftarrow \mathbb{Z}_p, \quad \ell \in [\log n] \ .$$

It then computes

$$\{I_\ell, B_\ell, A_\ell\}_{\ell=1}^{\log n}$$

and

$$\{z_\ell, w_\ell, v_\ell\}_{\ell=1}^{\log n}$$

according to the protocol description. For $\ell \in \{1, \ldots, \log n - 1\}$, it then picks random $U_\ell, R_\ell \leftarrow \mathbb{Z}_p$ and computes

$$D_\ell := \mathsf{Enc}_{\mathsf{pk}}(U_\ell; R_\ell) \ .$$

It then randomly picks $R \leftarrow \mathbb{Z}_p$, computes

$$D_0 := \frac{\mathsf{Enc}_{\mathsf{pk}}(0; R)}{\prod_{j=0}^{n-1} \left((C_j)^{x^{\log n}} \mathsf{Enc}_{\mathsf{pk}}(-\prod_{\ell=1}^{\log n} z_{\ell,j_\ell}; 0)\right)^{y^j} \cdot \prod_{\ell=1}^{\log n-1} (D_\ell)^{x^\ell}}$$

After that, Sim outputs the simulated transcript as

$$\left(\{I_\ell, B_\ell, A_\ell\}_{\ell=1}^{\log n}, y, \{D_\ell\}_{\ell=0}^{\log n-1}, x, \{z_\ell, w_\ell, v_\ell\}_{\ell=1}^{\log n}\right) .$$

This concludes our proof. □

3.8 Summary

In this chapter, we described our proposed treasury system for supporting decentralised cryptocurrency development. This is a really important step towards answering our research question and fulfils our second objective. Specifically, it enables us to provide an affirmative answer to our research question of designing provably secure self-sustenance mechanism for blockchain development through community collaboration. The proposed system leverage insights from our exploration of current blockchain practices in the industry and literature presented in Chapter 2.

We explain our design choices and provide a robust funding source for the treasury to support community proposals. Additionally, we abstract the underlying blockchain and explained how the different entities in our treasury system leverage the blockchain. Moreover, we formally discuss crucial protocols e.g., the distributed key generation protocol, that make up the voting scheme at the core of our treasury system.

Given the distributed architecture of our voting scheme, we provide details of voter/-expert registration and voting committee selection. The voting scheme supports liquid democracy by allowing voters to either delegate their votes to experts or vote directly on proposals in the treasury system.

The integrity of the voting tally and privacy of voter and expert ballots are guaranteed as long as a majority of the voting committee members are honest. However, in the

unlikely scenario where the majority (or all) committee members are corrupted by an adversary, the integrity of the election tally is still guaranteed using the zero-knowledge proofs publicly available on the blockchain. Although, the ballot secrecy of all voters and experts can is no longer guaranteed.

Furthermore, we provide the formal description of our novel zero-knowledge proofs for unit vector ballot with log size communication. This helps improves efficiency when compared with similar schemes e.g. Helios, that require proof size that is linear in the size of the ballot.

However, the current scheme supports blockchains where the amount of stake of any stakeholder is publicly available on the public distributed ledger. Therefore, we provide an extension of the techniques and protocols in Chapter 4 to support blockchains with private stake i.e. private voting power/weight.

Chapter 4

Privacy-Preserving Blockchain Decision-making and Consensus Evaluation

4.1 Overview

Closely related to treasury system operations on distributed ledger technologies is blockchain governance. Governance is mainly concerned with the process of establishing and maintaining order within systems (comprising agents with potentially conflicting interests) to ensure continued assured mutual benefits. Generally, IT governance is defined as a framework for decision rights and accountabilities to promote helpful conduct in the IT utilisation [MZZS18]. Specifically, within the context of blockchains, while principally, a treasury system is focused on decentralised funding of blockchain projects towards self-sustenance for maintenance and development [ZOB18, KKN$^+$], blockchain governance is mostly concerned with the procedures to approving or making protocol changes that affect blockchain consensus rules.

Despite the profound enthusiasm and promise of disruptive innovation around blockchain technologies, there remain questions around coordination and governance of activities on the blockchain, i.e. who is in charge? [ZPMS19]. Although, related to

free open-source software operations, a crucial goal of governance in blockchains is to prevent adverse forks (which are largely cheap to carry out on open-source software) from disagreements.

It is well documented that disagreement within communities of blockchain technologies (cryptocurrencies) stakeholders and practitioners can result in forks of the systems, thereby, leading to severe weakening, diminished exchange value and utility, etc. of those systems [DFL16, DuP17]. Real-world examples of such cryptocurrency forks include Bitcoin Cash, Bitcoin Gold, Ethereum Classic, etc.

The research presented in this chapter mainly represents non-trivial extension of the concepts, techniques, and model of the blockchain-based treasury system presented in Chapter 3. Specifically, the collaborative blockchain-based treasury system in Chapter 3 finds application in blockchain cryptocurrencies with public stake distribution. Namely, the treasury systems assumes that information about the amount of stake/coins owned by all stakeholders is publicly available on the public ledger. Note that, the public availability of stake distribution does not imply that identity of stakeholders is also public, i.e. identities could be pseudonymous, e.g. tied to addresses which are (hash of) public keys.

The Bitcoin blockchain represents a good example of such a blockchain protocol because information about the amount of Bitcoins owned by any address can be obtained from publicly available data on the blockchain [WHO19] whilst the identity of the owner is not revealed by the blockchain. However, we note that there exists several heuristics and techniques (e.g. transaction graph analysis, network de-anonymisation) to attack and compromise privacy and anonymity on public blockchains. The interested reader is referred to [CELR18] for a detailed survey on the topic.

The public availability of transaction information on the ledger resulted in privacy and security issues in first generation blockchain protocols. Consequently, privacy enhancing-techniques such as stealth addresses [NBF+16], coin mixing, coinjoin [Max13] were introduced to Bitcoin. Additionally, entirely new privacy-centric blockchain protocols e.g. Zerocash [BCG+14] and CryptoNote [vS13] were also created to address the privacy

issues of the Bitcoin protocol. Recently, for the first time [KKKZ19] provided a secure PoS-based blockchain protocol that securely realises a distributed ledger under the UC framework. The PoS-based privacy-preserving protocol of [KKKZ19] rely on combinations of new cryptographic primitives key-private forward-secure encryption and evolving coins to achieve security while Zerocash [BCG$^+$14] rely on 'transaction pouring' through zero-knowledge Succinct Non-interactive ARguments of Knowledge (zk-SNARKs).

Meanwhile, our new general decision-making system for blockchain governance provides a means for reaching resolutions and supports cryptocurrencies with private stake information where amount of stake owned by any stakeholder is not publicly available on the open ledger e.g. Zcash, Monero. Beyond supporting general blockchain decision-making, it is compatible with most existing off-the-shelf cryptocurrencies and is easily adaptable to a variety of decision-making rules, e.g. simple majority, fuzzy threshold voting. For instance, the approval voting rule in the treasury system can easily be extended to accommodate beyond the three main options- 'YES, NO, ABSTAIN' - based on the issues being decided upon.

Furthermore, to accommodate private stake information, in addition to the novel unit vector proof in Section 3.7, a new honest verifier zero-knowledge (HVZK) proofs/arguments for the multiplicative relation between two vector ciphertexts is provided in Section 4.3. In essence, it is a σ-protocol for making proofs that the contents of some commitments have a multiplicative relationship with each other, i.e. a zero-knowledge proof of knowledge to demonstrate that two ciphertexts satisfy a multiplicative relationship.

Liquid democracy (also known as delegative democracy [For02]) as a hybrid of direct democracy and representative democracy provides the benefits of both system [GA15, CMM$^+$16, KMP18] by enabling organisations to take advantage of the experts in a voting process and also gives every member the opportunity to vote [ZZ17a, ZZ17b]. Green in [GA15] axiomatically analysed that a delegation system is more democratic than traditional representative or a direct voting system. Although the advantages of liquid democracy has been widely discussed in the literature [Lom03, BBCV09, Nor03, GA15, Ito04], there are few provably secure construction of liquid democracy voting.

Most real-world implementations of liquid democracy only focus on the functionality aspect of their schemes. For instance, Google Vote [HL15] is an internal Google experiment on liquid democracy over the social media, Google+, which does not consider voter privacy. Similarly, systems such as proxyfor.me [Pro15], LiquidFeedback [Liq], Adhocracy [Ad], GetOpinionated, [Deg] also offer poor privacy guarantees. It is worth mentioning that Sovereign [Dem17] is a blockchain-based voting protocol for liquid democracy; therefore, its privacy is inherited from the underlying blockchain, which provides pseudonymity-based privacy. Wasa2il [McC16] is able to achieve end-to-end verifiability because this foils privacy. The best known liquid democracy and proxy democracy voting schemes are nVotes [nVo17] and Statement Voting [ZZ17a, ZZ17b]. However, those systems require mix-nets as their underlying primitive. This makes them less compatible to the blockchain setting due to the heavy work load of the mixing servers.

There are a few blockchain based e-voting schemes in the literature, but most of them, e.g. Agora [Ago18], only use the blockchain as a realization of the public bulletin board. The actual e-voting schemes are not integrated with the blockchain. Lee et. al., in [LJEK16] proposed a blockchain-based voting solution that heavily relies on an external 'trusted third party' between users and the election authority/authentication authority, in order to ensure anonymity/privacy of voters. Each candidate is voted for by having transactions sent to them. Nonetheless, privacy or anonymity of voters can be broken by collusion between the authentication organisation and the trusted third party. Bistarelli et. al., in [BMSS17] proposed an end-to-end voting system based on Bitcoin that utilises a Kerberos-based protocol to achieve voter identity anonymisation. Voting takes place via sending of tokens from voters to address (public key) of candidates. However, voting is not private and other voters can be influenced by the trend or likelihood of the overall results (before voting is concluded). Furthermore, the scheme is susceptible to coercion. Our work differs from these earlier works because it supports liquid democracy whilst preserving privacy of the voters and delegates involved in the decision-making process.

The remainder of this chapter is organised as follows. In Section 4.2, we present the polling system for decision-making in Section 4.2 and a high-level description of the

main extensions to the scheme presented in Subsection 3.6.4. Next, we provide the new multiplicative relation zero-knowledge proof in 4.3. Finally, an examination of the quality of individual choices with respect to final polling decisions is presented in Section 4.4. The analysis serves as 'soft-consensus' evaluation of the general blockchain decisions by enabling individual participants and the entire system to evaluate agreement on issues based on decision outcomes. Consequently, this encourages increased discussions, interactions and general understanding and context among community members towards issues being decided upon within the system.

4.2 The Proposed Polling Scheme

Here, we briefly describe the main differences between the original scheme in the proposed treasury system and the privacy-preserving scheme covered in this chapter. Specifically, two main changes are required to support blockchains with private stake. The first change involves the voter ballot and the other change covers the process of computing the voting tally at the end of voting. Next, we recall the ballot encoding process from the treasury system voting scheme below and thereafter explain the change in the ballot creation process.

Let m be the number of experts and n be the number of voters. Let $\mathbf{e}_i^{(m)} \in \{0,1\}^m$ be the unit vector where its i-th coordinate is 1 and the rest coordinates are 0.We use $\mathsf{Enc}_{\mathsf{pk}}(\mathbf{e}_i^{(\ell)})$ to denote coordinate-wise encryption of $\mathbf{e}_i^{(\ell)}$, i.e. $\mathsf{Enc}_{\mathsf{pk}}(e_{i,1}^{(\ell)}), \ldots, \mathsf{Enc}_{\mathsf{pk}}(e_{i,\ell}^{(1)})$, where $\mathbf{e}_i^{(\ell)} = (e_{i,1}^{(\ell)}, \ldots, e_{i,\ell}^{(\ell)})$. Let $\mathbf{u}_i^{(\ell)} = \mathsf{Enc}_{\mathsf{pk}}(\mathbf{e}_{i,1}^{(\ell)})$

In the treasury system, a voter's ballot is represented as $\mathsf{Enc}_{\mathsf{pk}}(\mathbf{e}_i^{(\ell)})$. However, in the privacy-preserving voting scheme, the original unit-vector ballot is accompanied with a second unit-vector ballot $\mathbf{v}_i^{(\ell)} = (v_{i,1}^{(\ell)}, \ldots, v_{i,\ell}^{(\ell)})$.

Basically, at a high-level, each voter provides a zero-knowledge proof that confirms that there exists a multiplicative relationship between $\mathbf{u}_i^{(\ell)}$, $\mathbf{v}_i^{(\ell)}$ and their stake α whose encrypted value can be verified on the public ledger. See Section 4.3 for a formal presentation of the multiplicative relation zero-knowledge proof. Therefore, the additional ballot $\mathbf{v}_i^{(\ell)}$ can be interpreted as a weighted version of the original ballot $\mathbf{e}_i^{(\ell)}$ as described

in the equation below.

$$\mathbf{v}_i^{(\ell)} = \alpha \cdot \mathbf{e}_i^{(\ell)}$$

Therefore, given the already weighted unit vector ballots, the tally procedure no longer requires the weighting of voter ballots before homomorphic addition. The tally procedure then proceeds as described in the original delegative voting scheme in Chapter 3. Specifically, the delegations section of each ballot is homomorphically combined and jointly decrypted by the voting committee using decryption shares they provide. Thereafter, expert ballots are weighted by the amount of delegation received and homomorphically combined with the *direct voting sections* of the voters' ballots to obtain the encrypted tally. Finally, voting committee members provide decryption shares and jointly decrypt the election tally ciphertext. Figure 4.1 depicts an overview of the main pre-voting, voting and post-voting operations in the privacy-preserving scheme. Basically, it depicts operations carried out by the voting committee, voters and experts in the election process. It covers the ballot casting procedures by voters and experts as well as the procedures involved in the election tally computation. Additionally, a formal presentation of the protocol for the polling scheme is depicted in Fig. 4.2.

4.3 A New Vector Multiplicative Relation Zero-knowledge Proof

Besides the novel unit vector zero-knowledge proof of Section 3.7, the multiplicative relation zero-knowledge protocol presented in this section are required for secure execution of the polling protocol $\mathcal{F}_{\text{POLL}}$ described in Fig. 4.2.

Let $\mathbf{u}^{(n)} = (u_0, \ldots, u_{n-1})$ and $\mathbf{v}^{(n)} = (v_0, \ldots, v_{n-1})$ be two vectors of n elements such that $v_i = \alpha \cdot u_i$ for $i \in [n]$. Given element-wise encryption of $\mathbf{u}^{(n)}$, $\mathbf{v}^{(n)}$ and α, we now construct a 3-move honest verifier ZK proof of knowledge showing that $\mathbf{u}^{(n)}$ and $\mathbf{v}^{(n)}$ are in the correct multiplicative relation with respect to α. Specifically, for $i \in [0, n-1]$, given $C := \mathsf{Enc}_{\mathsf{pk}}(\alpha; \beta)$ and $A_i := \mathsf{Enc}_{\mathsf{pk}}(a_i; r_i)$, we set $V_i := C^{a_i} \cdot \mathsf{Enc}_{\mathsf{pk}}(0; t_i)$. This allows the

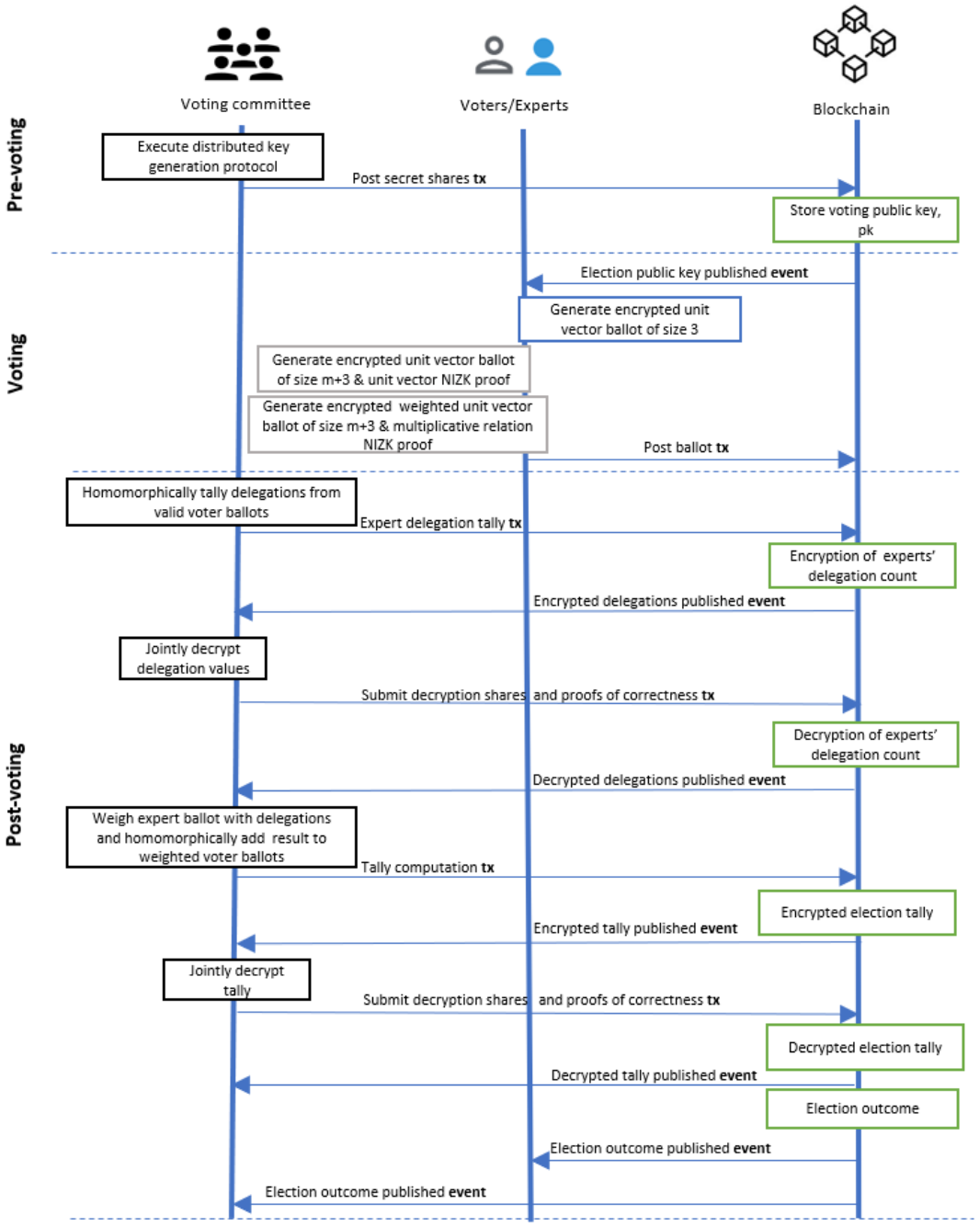

Figure 4.1: An overview of interactions among voting committee members, voters, experts and the underlying blockchain in the privacy-preserving delegable voting scheme for general blockchain-based decision-making.

The voting protocol $\Pi_{\text{POLL}}^{t,k,m,n}$

Setup phase:

- Upon receiving $(\text{INIT}, \mathsf{sid})$ from the environment $\mathcal{Z}$, the committee C_j, $j \in [k]$ sends $(\text{KEYGEN}, \mathsf{sid})$ to $\mathcal{F}_{\text{DKG}}^{t,k}$ to generate pk.

Ballot casting phase:

- Upon receiving $(\text{VOTE}, \mathsf{sid}, v_j)$ from the environment $\mathcal{Z}$, the expert E_j, $j \in [m]$ does the following:
  - Send $(\text{READPK}, \mathsf{sid})$ to $\mathcal{F}_{\text{DKG}}^{t,k}$, and receive $(\text{PUBLICKEY}, \mathsf{sid}, \mathsf{pk})$ from $\mathcal{F}_{\text{DKG}}^{t,k}$.
  - Set the unit vector $\mathbf{e}^{(3)} \leftarrow \mathsf{encode}^{\mathsf{E}}(v_j)$. Compute $\mathbf{c_j}^{(3)} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathbf{e}^{(3)})$ and its NIZK proof π_j
  - Post $(\mathsf{E}_j, \mathbf{c_j}^{(3)}, \pi_j)$ to $\mathcal{F}_{\text{LEDGER}}$.
- Upon receiving $(\text{CAST}, \mathsf{sid}, v_i, \alpha_i)$ from the environment $\mathcal{Z}$, the voter V_i, $i \in [n]$ does the following:
  - Send $(\text{READPK}, \mathsf{sid})$ to $\mathcal{F}_{\text{DKG}}^{t,k}$, and receive $(\text{PUBLICKEY}, \mathsf{sid}, \mathsf{pk})$ from $\mathcal{F}_{\text{DKG}}^{t,k}$.
  - Set the unit vector $\mathbf{e}^{(m+3)} \leftarrow \mathsf{encode}^{\mathsf{V}}(v_i)$. Compute $\mathbf{c_i}^{(m+3)} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathbf{e}^{(m+3)})$ and its NIZK proof σ_i
  - Compute $\hat{\alpha}_i \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\alpha_i)$ and $\mathbf{v_i}^{(m+3)} := ((\hat{\alpha}_i)^{e_0} \cdot \mathsf{Enc}_{\mathsf{pk}}(0), \ldots, (\hat{\alpha}_i)^{e_{m+2}} \cdot \mathsf{Enc}_{\mathsf{pk}}(0))$ together with a NIZK proof ρ_i.
  - Post $(\mathsf{V}_i, \mathbf{u_i}^{(m+3)}, \mathbf{v_i}^{(m+3)}, \sigma_i, \hat{\alpha}_i, \rho_i)$ to $\mathcal{F}_{\text{LEDGER}}$.

Tally/Result phase:

- Upon receiving $(\text{DELCAL}, \mathsf{sid})$ from the environment $\mathcal{Z}$, the committee C_t, $t \in [k]$ does:
  - Read $\mathcal{F}_{\text{LEDGER}}$ and obtain data.
  - Fetch the ballots $\{(\mathsf{E}_i, \mathbf{c_i}^{(3)}, \pi_i)\}_{i\in[m]}$ and $\{(\mathsf{V}_i, \mathbf{u_i}^{(m+3)}, \mathbf{v_i}^{(m+3)}, \sigma_i, \hat{\alpha}_i, \rho_i)\}_{i\in[n]}$ from data.
  - For $i \in [m]$, check $\mathsf{Verify}(\mathbf{c_i}^{(3)}, \pi_i) = 1$; for $j \in [n]$, $\mathsf{Verify}(\mathbf{u_j}^{(m+3)}, \sigma_j) = 1$ and $\mathsf{Verify}(\hat{\alpha}_i, \mathbf{u_j}^{(m+3)}, \mathbf{v_j}^{(m+3)}, \rho_j) = 1$. Remove all the invalid ballots.
  - For $j \in [n]$, if a valid $\mathbf{v_j}^{(m+3)}$ is posted, parse $\mathbf{v_j}^{(m+3)}$ to $(\mathbf{a_j}^{(m)}, \mathbf{b_j}^{(3)})$.
  - For $i \in [0, m-1]$, compute $s_i := \prod_{\ell=1}^{n} a_{\ell,i}$ and jointly decrypt it to w_i
- Upon receiving $(\text{TALLY}, \mathsf{sid})$ from the environment $\mathcal{Z}$, the committee C_t, $t \in [k]$ does:
  - For $i \in [0, m-1]$, $\ell \in [0, 2]$, compute $d_{i,\ell} := c_{i,\ell}^{w_i}$.
  - For $\ell \in [0, 2]$, compute $x_\ell := \prod_{j=0}^{m-1} d_{j,\ell} \cdot \prod_{j=1}^{n} b_{j,\ell}$ and jointly decrypt it to y_ℓ
- Upon receiving $(\text{READTALLY}, \mathsf{sid})$ from the environment $\mathcal{Z}$, the party P does the following:
  - Read $\mathcal{F}_{\text{LEDGER}}$ and obtain data.
  - Fetch $\{(x_i, y_i)\}_{i\in[0,2]}$ from data, and return $(\text{READTALLYRETURN}, \mathsf{sid}, (y_0, y_1, y_2))$ to the environment $\mathcal{Z}$.

Figure 4.2: The voting protocol $\Pi_{\text{POLL}}^{t,k,m,n}$ in $\{\mathcal{F}_{\text{LEDGER}}, \mathcal{F}_{\text{DKG}}^{t,k}\}$-hybrid model

prover to be oblivious about the plaintext inside C to complete the proof. The protocol is depicted in Fig. 4.3. Details of implementation performance metrics e.g. running time and proof size, are provided in Chapter 5.

Vector multiplicative relation ZK argument

Statement: pk, C, $\{A_i := \mathsf{Enc}_{\mathsf{pk}}(a_i; r_i)\}_{i=0}^{n-1}$ and $\{V_i := C^{a_i} \cdot \mathsf{Enc}_{\mathsf{pk}}(0; t_i)\}_{i=0}^{n-1}$

Witness: $\{a_i, r_i, t_i\}_{i=0}^{n-1}$

Protocol:

- The prover P:
  - Pick random $x, y, z \leftarrow \mathbb{Z}_p$;
  - Compute $X := \mathsf{Enc}_{\mathsf{pk}}(x, y)$ and $Z := C^x \cdot \mathsf{Enc}_{\mathsf{pk}}(0; z)$;
- $P \rightarrow V$: X, Z;
- $V \rightarrow P$: Random $\rho \leftarrow \{0,1\}^\lambda$;
- The prover P does:
  - Set $x' := x + \sum_{i=0}^{n-1} a_i \cdot \rho^{i+1}$, $y' := y + \sum_{i=0}^{n-1} r_i \cdot \rho^{i+1}$, and $z' := z + \sum_{i=0}^{n-1} t_i \cdot \rho^{i+1}$;
- $P \rightarrow V$: x', y', z';

Verification:

- Return valid if and only if the following are true:
  - $Z \cdot \prod_{i=0}^{n-1} V_i^{\rho^{i+1}} = C^{x'} \cdot \mathsf{Enc}_{\mathsf{pk}}(0; z')$
  - $X \cdot \prod_{i=0}^{n-1} A_i^{\rho^{i+1}} = \mathsf{Enc}_{\mathsf{pk}}(x', y')$

Figure 4.3: Vector multiplicative relation ZK argument

Theorem 2. *The protocol described in Fig. 4.3 is a* 3*-move public coin special honest verifier zero-knowledge argument of knowledge of* $\{a_i, r_i, t_i\}_{i=0}^{n-1}$ *such that:*

$$\{A_i := \mathsf{Enc}_{\mathsf{pk}}(a_i; r_i)\}_{i=0}^{n-1}$$

and

$$\{V_i := C^{a_i} \cdot \mathsf{Enc}_{\mathsf{pk}}(0; t_i)\}_{i=0}^{n-1} \ .$$

Proof. For perfect completeness, we first observe that the following verification equations hold:

$$Z \cdot \prod_{i=0}^{n-1} V_i^{\rho^{i+1}} = C^{x'} \cdot \mathsf{Enc}_{\mathsf{pk}}(0; z')$$

and

$$Y \cdot \prod_{i=0}^{n-1} A_i^{\rho^{i+1}} = \mathsf{Enc}_{\mathsf{pk}}(x', y') \ .$$

Indeed, by additively homomorphic property,

$$\begin{aligned} Z \cdot \prod_{i=0}^{n-1} V_i^{\rho^{i+1}} &= C^{x} \cdot \prod_{i=0}^{n-1} (C^{a_i})^{\rho^{i+1}} \cdot \mathsf{Enc}_{\mathsf{pk}}(0; z) \cdot \prod_{i=0}^{n-1} \mathsf{Enc}_{\mathsf{pk}}(0; t_i)^{\rho^{i+1}} \\ &= C^{x + \sum_{i=0}^{n-1} a_i \cdot \rho^{i+1}} \cdot \mathsf{Enc}_{\mathsf{pk}}(0; z + \sum_{i=0}^{n-1} t_i \cdot \rho^{i+1}) \\ &= C^{x'} \cdot \mathsf{Enc}_{\mathsf{pk}}(0; z') \ . \end{aligned}$$

Similarly,

$$X \cdot \prod_{i=0}^{n-1} A_i^{\rho^{i+1}} = \mathsf{Enc}_{\mathsf{pk}}(x, y) \cdot \prod_{i=0}^{n-1} \mathsf{Enc}_{\mathsf{pk}}(a_i \cdot \rho^{i+1}; r_i \cdot \rho^{i+1}) = \mathsf{Enc}_{\mathsf{pk}}(x', y') \ .$$

For soundness, we show that the protocol is an argument of knowledge (AoK), by showing that it has a witness-extended emulator. Since the challenge $\rho \in \mathbb{Z}_p$ is randomly chosen, by Schwartz-Zippel lemma, the prover has negligible probability of convincing the verifier unless all ρ^i related terms match on each side of the equality for all $i \in \{0, \ldots, n-1\}$. The witness-extended emulator Ext runs $\langle P^*, V \rangle$ to get a transcript.

If the prover P^* has probability $p(\lambda)$ of making an acceptable argument, the black-box witness-extended emulator Ext also has success probability $p(\lambda)$ to produce an accepting argument. It rewinds the protocol to the challenge phase and runs it with fresh challenges until it has n acceptable arguments. Since the prover P^* has probability $p(\lambda)$ of making an accepting argument in the first place, the emulator Ext will take an average of $n/p(\lambda)$ rewinds, which takes $\mathsf{poly}(\lambda)$ running time. Again, there is overwhelming probability that we have transcripts with n different challenges. The n different challenges give us a $n \times n$ transposed Vandermonde matrix

$$\mathcal{V} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ \rho_1 & \rho_2 & \dots & \rho_n \\ \vdots & \vdots & \ddots & \vdots \\ \rho_1^{n-1} & \rho_2^{n-1} & \dots & \rho_n^{n-1} \end{bmatrix}$$

Note that $\mathcal{V}$ is invertible because $\rho_1, \dots, \rho_n$ are different, so Ext can compute $\mathcal{V}^{-1}$. Let x'_i, y'_i, z'_i denote the prover's replay when ρ_i is challenged in the accepting transcripts. By concatenating x'_i, y'_i, z'_i, we can get three vectors

$$\mathcal{X} := (x'_1, \dots, x'_n), \mathcal{Y} := (y'_1, \dots, y'_n), \mathcal{Z} := (z'_1, \dots, z'_n) \ .$$

Hence, Ext can compute

$$(x, u_0, \dots, u_{n-1}) := \mathcal{X} \cdot \mathcal{V}^{-1},$$

$$(y, r_0, \dots, r_{n-1}) := \mathcal{Y} \cdot \mathcal{V}^{-1},$$

and

$$(z, t_0, \dots, t_{n-1}) := \mathcal{Z} \cdot \mathcal{V}^{-1}$$

Thereafter, Ext outputs the witness $\{a_i, r_i, t_i\}_{i=0}^{n-1}$.

For perfect special honest verifier zero-knowledge, we need to construct a simulator $\mathcal{S}$, that on receiving challenge ρ outputs the simulated argument that is indistinguishable from a real argument with challenge ρ. On challenge ρ, the simulator $\mathcal{S}$ randomly selects

$x', y', z' \leftarrow \mathbb{Z}_p$. It then computes

$$X := \mathsf{Enc}_{\mathsf{pk}}(x', y') / \prod_{i=0}^{n-1} A_i^{\rho^{i+1}}$$

and

$$Z := C^{x'} \cdot \mathsf{Enc}_{\mathsf{pk}}(0; z') / \prod_{i=0}^{n-1} V_i^{\rho^{i+1}}$$

The simulator $\mathcal{S}$ then outputs (X, Z, ρ, x', y', z'). Clearly, the distribution of simulated transcript is identical to the distribution of a real one. Hence, the protocol is perfect special honest verifier zero-knowledge. □

4.4 Analysing Consensus

The overarching aim of decision-making mechanisms is to reach the best decision. However, it is usually unclear what constitutes the best alternative. In other words, in a multi-party decision-making process, it is difficult to agree on what constitutes the best solution, due to differences in individual preferences, interests, knowledge, skill, orientation, etc. Therefore, integration of community-wide knowledge, skills and expertise of members is fundamental for long-term sustainability of jointly-maintained utilities such as blockchain (developmental projects).

Consensus building [IB99] has been proposed as a way to deal with complex, strategic and often controversial planning and decision-making. Sustained innovation and development requires the continued maintenance of the complex interaction between all stakeholders (with varying expertise, skill sets and values). The goal is to select and implement solutions that offers 'mutual' gains among contending, and potentially conflicting stakeholders. Within collaborative governance, consensus emanates from inclusive representation of all relevant participants with shared objective, and commitment to providing solutions and establishing and building trust towards marginal successes [IB99, ENB11, AG07].

Consensus building or processes are typically time-costly and resource-consuming.

However, the nature of decentralised blockchain infrastructure helps mitigate against these potential challenges. For instance, by design, planning on decision-making within the governance and treasury system supports lengthy discussions and deliberations spanning a range of system epochs in any given system period. Specifically, there is ample time to discuss and deliberate on submissions, issues and proposals before the voting process. Similarly, the decentralised nature of blockchain technologies and 'electronic' voting reduces required resources associated with general decision-making, e.g. access to web browser and internet [Adi08].

Conversely, within collaborative decision-making settings, it is suggested that consensus kills innovation, creativity and uniqueness by encouraging individuals that participate in the process to (tend to) abandon their decisions and align with the group decision. However, this is not necessarily true considering that the debates and discussions take place before compromises are made or decisions are reached. Moreover, our decision-making systems provide vote privacy because ballots are encrypted. Hence, it is difficult for any individual participant to pre-emptively align their votes with others (or the group choice) because it is not clear what other participants' choices or the most popular choices are (before tally results are released). Accordingly, rather than suppress the individual, consensus empowers the individual through social activity and interaction [Tri89], and can lead to novel practices, ideas, and relationships within the community.

Research evidence shows that engaged citizenry/community is better than a passive one [IS04] and collaborative decision-making with community-inclusive participation can also help achieve improved community relations. As a result, participants become willing evaluators of decisions and policies, which results in improved community-wide support for decisions reached rather than factional support. Thus, enhancing the effectiveness of collaborative decision-making for governance, which is particularly important to maintaining sustainable blockchain systems.

Agreement represents, perhaps, the single most popular criterion for evaluating consensus. Typically in the literature, consensus is analysed in terms of agreement [IB99, ENB11,BM10]. We emphasise that consensus is not *Majority Rule*. Consensus involves the

evaluation of agreement among a set of parties on a set of alternative solutions [HVHC02] in a multi-party collaborative decision-making process (e.g. blockchain decision-making process). Classical definitions of consensus imply absolute agreement among all parties as a condition for consensus [IB99]. This type of (full) consensus is quite feasible within small teams or organisations with members having relevant information needed for decision-making, although, it allows a single 'opposing' member to stall the process of decision-making. Moreover, the extreme implausibility of guaranteed/continuous full agreement on all issues among parties, makes this definition of consensus problematic and less useful. By extension, this definition would equate the utility of decisions with 'almost unanimous', i.e. very high but not perfect, agreement to those with complete disagreement - as both being not useful. Nonetheless, in real-world scenarios multi-party decisions need not be unanimous or in absolute agreement (full consensus) for decisions to be useful.

Soft consensus or acceptable threshold of consensus (rather than full consensus) suffices for collaborative decision-making systems involving dispersed and cross-functional networks/participants [IH18]. We remark that setting thresholds for consensus is difficult due to the inherent dynamic nature of consensus and the issues/topics being decided upon. No two decision-making scenarios are the same. In evaluating consensus, it is critical to identify the uniqueness and nuances of each decision-making scenario. For instance, any real-world assessment of consensus of a collaborative decision-making process need to consider the nature of the decision, the open and available facts, the issues of contention, amount of funds involved (in the case of treasury), proposal submission and deliberation time, participants, etc., for any useful inference to be drawn about understanding and improving consensus. Identifying consensus building processes and outcomes is difficult and effective consensus requires adaptation and can sometimes be autonomous [IB99].

Therefore, in order to accommodate the spectrum of consensus between unanimous agreement and total disagreement, soft agreement is defined as an iterative dynamic process that evaluates the agreement between all participants, and the agreement between the individual participant's preference and the group solution [HVHC02]. Typically, two

primary related measures are used to evaluate consensus. Namely, *Consensus measure* - measure of agreement among all participants - and *Proximity measure* - measure of agreement between individual solutions and collective solution [HVHC02, IB99].

In line with the aim of collaborative decision-making, we note that our key goal of evaluating consensus is not to produce 'winners and losers', rather, the goal is to build, enhance and encourage community-wide participation and acceptance (sense of belonging and responsibility) and ownership of the growth, changes and developments of the underlying blockchain system. Consequently, feedback is a key component of consensus evaluation because information obtained from proximity measure is useful for influencing discussion and minimising disagreement among stakeholders [HVHC02].

Evidently, developments or changes with better consensus are more durable and sustainable because high consensus implies a higher agreement (support) among the parties of the decision-making process. Furthermore, agreements of this nature tend to be of very high quality because they generally take into consideration the knowledge offered by each participant [IB99] rather than only interests of some parties.

With the help of an illustrative example of a system period, we now provide an evaluation of consensus of our blockchain-based decision-making process.

4.4.1 Example of Consensus Evaluation

Typically, consensus is measured through the use of some dissimilarity function e.g. cosine of angles, Euclidean distance. The dissimilarity is measured between corresponding individual preferences/solutions (proximity measure) and group solution, as well as the evaluation of agreement among all participants on the group solution (consensus measure).

For the purpose of consensus measurement in our blockchain decision-making system, we propose an adaptation of the approach of [BM10], which itself is an adaptation of [HVHC02]. In [BM10], proximity measure and consensus measure were utilised in evaluating the degree of consensus on a spatial-GIS decision among relevant stakeholders. Our adaptation is essential in order to accommodate subtleties peculiar to blockchain

systems (or cryptocurrencies) e.g. cryptocurrency stake distribution. Specifically, for consensus measurement on our blockchain-based decision-making system, we identify the following key elements:

- Proposals, issues or alternative solutions
- Stakeholders or participants in the decision-making process
- Available resources (or system funds)
- The decision-making process (or voting scheme)
- Outcomes of the decision-making process (or solution set)
- Agreement among all participants (or consensus measure)
- Agreement between individual solutions and the system outcome/solution (or proximity measure)

Furthermore, below is a list of the processes/steps for consensus evaluation (of decisions) within our blockchain-based collaborative decision-making system:

- Preference specification
- Collective solution calculation
- Distance measure
- Distance aggregation
- Consensus measure and
- Proximity measure

We now present an explanation of each of the above-outlined steps involved in the consensus evaluation process using 10 hypothetical example proposals that have been submitted to the blockchain and decided upon using the decision making system.

Following results in usability [Vir92, Lew94], we assume 5 participants (2 experts and 3 voters) are involved in the current voting process. For simplicity, without the loss of

generality, we also assume a flat model of stake distribution in this illustrative period of the decision-making system. Namely, we assume that all participants (expert/voter) have equal stake in the system e.g. 1 coin each.

4.4.1.1 Preference Specification

Each participant respectively specify his preferences based on his assessment of the individual proposals, using criteria/guidelines such as: usefulness of proposal, timeliness, cost-benefit impact of proposal, profile of proposer, relevance of proposal, urgency of proposal, amount of funds requested, duration of proposal, quality of proposal, etc. However, users are free to further evaluate proposals as they deem fit (or based on their personal judgement).

Particularly, users vote YES, NO, ABSTAIN for proposals either directly or indirectly by delegating their voting power to experts in that particular topic. For consensus evaluation, we encode a YES, NO, *or* ABSTAIN vote as $1, 0$, or $\perp$ respectively. We remark that the ballots of users who vote ABSTAIN ($\perp$) for any proposal are treated as being the same as the system outcome for that proposal. Hence, for consensus evaluation, they are considered as being in agreement with whatever is the outcome of the group decision-making for the affected proposal.

4.4.1.2 Collective Solution Calculation

The group solution of our decision-making system is obtained through the application of the voting rule on all the ballots cast by the experts and voters in the system. Specifically, for our system, where the voting rule is *Fuzzy Threshold Voting*, this corresponds to ranking of the alternative proposals based on the number of votes for minus the number of votes against, and checking that the remainder is at least 10% of all votes recorded (or system threshold).

Particularly, in the case of treasury system voting, winning proposals are determined as those that receive funding from the (ranked) list of all 'qualified proposals'. Therefore, proposals that meet the minimum threshold but do not receive funding because of the

limitation of available funds are not considered members of the set of 'winning proposals'. The set of 'winning proposals' are those that satisfy all the winning requirements within the decision-making system.

Details of how the participants voted on all 10 proposals in our illustrative example is presented below. Table 4.1 provides information on how all voters and experts cast their ballots.

Table 4.1: User preference

	Prj 1	Prj 2	Prj 3	Prj 4	Prj 5	Prj 6	Prj 7	Prj 8	Prj 9	Prj 10
User 1 / Voter 1	1	B	1	B	1	B	A	A	B	$\perp$
User 2 / Voter 2	$\perp$	1	1	$\perp$	0	1	1	0	$\perp$	1
User 3 / Expert A	1	1	1	0	0	1	0	0	1	0
User 4 / Expert B	1	1	0	1	0	1	0	1	1	0
User 5 / Voter 3	0	1	A	1	$\perp$	A	1	$\perp$	A	1
Total	3	5	4	3	1	5	2	1	4	2
System Decision	0	1	1	0	0	1	0	0	1	0

4.4.1.3 (Normalised) Distance Measure

For each participant, we calculate the distance measure (DM) of every vote/ballot for each proposal by comparing the participant's choice with the system decision. We apply the dissimilarity function given below:

$$DM_{i,j} = |PP_{i,j} - TS_j|$$

$$NDM_{i,j} = \frac{DM_{i,j}}{|PS|}$$

where $DM_{i,j}$ (respectively, $NDM_{i,j}$) is the distance (respectively, normalised distance) between the i^{th} participant's choice for project j and the system solution for proposal j is TS_j. $PP_{i,j}$ is the i^{th} *participant's preference/choice* for project j and PS is the *preference size* of voter's choice.

In the case of our blockchain decision-making system $PS = 2$, for choices YES and NO because ABSTAIN is interpreted differently. As earlier explained, for consensus evaluation, the choice of voters/experts who vote ABSTAIN i.e. $\perp$ for any proposal is considered

as being the same as system decision for that particular proposal. Hence, a distance measure of zero (0) is assigned for a participant who votes ABSTAIN for any project proposal. The distance measure based on the users' choice specification for the system decision-making is presented in Table 4.2.

Table 4.2: Distance specification

	Prj 1	Prj 2	Prj 3	Prj 4	Prj 5	Prj 6	Prj 7	Prj 8	Prj 9	Prj 10
Voter 1	1	0	0	1	1	0	0	0	0	0
Voter 2	0	0	0	0	0	0	1	0	0	1
Expert A	1	0	0	0	0	0	0	0	0	0
Expert B	1	0	1	1	0	0	0	1	0	0
Voter 3	0	0	0	1	0	0	1	0	0	1

4.4.1.4 Distance Aggregation and Consensus Degree on Projects

Using the normalised distance measure, NDM, we calculate the degree of consensus among all participants (voters and experts) on each project as follows:

$$CD_j = 1 - \sum_{i=1}^{p} \frac{NDM_{i,j}}{p}$$

where CD_j is the consensus degree for project j, and p is the size/number of participants. The consensus degree based on the distance measure of users' choice in the system decision-making is presented in Table 4.3.

Table 4.3: Consensus degree

	Prj 1	Prj 2	Prj 3	Prj 4	Prj 5	Prj 6	Prj 7	Prj 8	Prj 9	Prj 10
Voter 1	0.5	0	0	0.5	0.5	0	0	0	0	0
Voter 2	0	0	0	0	0	0	0.5	0	0	0.5
Expert A	0.5	0	0	0	0	0	0	0	0	0
Expert B	0.5	0	0.5	0.5	0	0	0	0.5	0	0
Voter 3	0	0	0	0.5	0	0	0.5	0	0	0.5
Total	1.5	0	0.5	1.5	0.5	0	1.0	0.5	0	1.0
Consensus degree	0.7	1.0	0.9	0.7	0.9	1.0	0.8	0.9	1.0	0.8

4.4.1.5 Consensus Measure

We now proceed to calculate the overall consensus among all participants (for our illustrative example through an aggregation of the consensus degrees, CD. The consensus measure, CM, is calculated through the aggregation procedure of [YF94], which utilises Ordered Weighted Average (OWA). The aggregation operator enables the consensus degrees on the 'winning proposals' have more importance or weight [HVHC02] in the aggregation procedure. The expression for CM is as follows:

$$CM = (1 - \beta) \cdot \sum_{i=1}^{n} \frac{CD_i}{n} + \beta \cdot \sum_{s=1}^{t} \frac{CDW_s}{t}$$

where CDW is the set of consensus degrees for 'winning proposals', t is its cardinality, n is the total number of projects, and $\beta \in [0, 1]$ is used to control the influence of consensus degree of the winning projects on the overall consensus measure in the decision-making system.

Evidently, higher values of β causes consensus degree of the winning proposals to highly influence the overall consensus measure. Typical values of β recommended in the literature are $0.7, 0.8$, and 0.9 [HVHC02, BM10]. We use a β value of 0.8 in our example to emphasize the importance of consensus degree among the participants on the winning projects. Clearly, the value of β at any given (system) period will be influenced and determined by a number of system factors, e.g. number of proposals, urgency and crucialness of proposal, previous β values. Table 4.4 shows the consensus measure for different values of β.

Table 4.4: Consensus measure for various values of β

β	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
CM	0.87	0.881	0.891	0.902	0.912	0.923	0.933	0.944	0.954	0.964	0.975

4.4.1.6 Proximity Measure of Participants

Here, we evaluate the proximity measure, PM, of each participant's voting preference to the collective system-wide decision by aggregating each participant's distance measure across all the proposals. Analogous to the calculation of consensus measure, we utilise OWA aggregation operator as follows:

$$PM_i = (1-\beta)\frac{\sum_{j=1}^{p}(1-NDM_{i,j})}{p} + \beta\Big(1 - \frac{\sum_{k=1}^{t} NDMW_{i,k}}{t}\Big)$$

where $NDM_{i,j}$ is the distance measure between participant $i's$ preference for proposal j and the collective decision for proposal j. For participant i, $NDMW_{i,k}$ is a special normalised distance measure between the collective decision for proposal k in the 'set of winning proposals' and the corresponding participant $i's$ preference for that proposal. Thus, NDMW only considers normalised distance measures for proposals that satisfy the decision-making system winning requirements, i.e. winning proposals.

Table 4.5: Proximity measure for $\beta = 0.8$

	Prj. 1	Prj. 2	Prj. 3	Prj. 4	Prj. 5	Prj. 6	Prj. 7	Prj. 8	Prj. 9	Prj. 10	Avg.	PM_i	$PM_i(\beta = 0.8)$
U1	0.5	1	1	0.5	0.5	1	1	1	1	1	0.85	$(1-\beta)0.85+\beta$	0.97
U2	1	1	1	1	1	1	0.5	1	1	0.5	0.9	$(1-\beta)0.9+\beta$	0.98
E.A	0.5	1	1	1	1	1	1	1	1	1	0.95	$(1-\beta)0.95+\beta$	0.99
E.B	0.5	1	0.5	0.5	1	1	1	0.5	1	1	0.8	$(1-\beta)0.8+0.875\beta$	0.86
U5	1	1	1	0.5	1	1	0.5	1	1	0.5	0.85	$(1-\beta)0.85+\beta$	0.97

As earlier explained, we assign a value of 0.8 to β and present the proximity measure between each participant's preference and the system decision in Table 4.5 above. Table 4.5 summarises the proximity measure for the example considered. Evidently, participants with high proximity measures contribute positively towards the system consensus while proximity measures close to zero signify negative contribution towards overall decision-making system consensus. Besides, it can be observed that Voter 1 and Voter 5 both have the same proximity measure of 0.97, despite having different voting preferences. This can be explained by the observation that these two voters voted exactly the same way for proposals in the 'winning set'. In turn, the 'winning set' or collective decision highly influenced our aggregated measures due to the high β value used.

4.5 Applications

In this section we briefly discuss additional independent applications for some of the delegable voting systems at the core of our treasury system presented in Chapter 4 and its extension discussed in Chapter 5. Specifically, we explore the use of the system in decentralised blockchain oracles and provide general guideline for their adoption by ERC20 token based blockchains on Ethereum.

For increased utility of blockchains, there is need for these platforms to provide data to and access data from sources that are external to the blockchains themselves [ZCC+16b]. Oracles are simply external/additional sources of data/information for a system e.g., hashing oracles, decryption oracles, etc. For blockchain systems, an oracle is an agent that provides additional real-world data that are external to the blockchain's system. Data oracles can be used to retrieve general data which may be needed for triggering certain conditions in smart contracts applications based on real-world events.

Decentralised oracles can be used as market data oracles for providing blockchains with data about real-world markets. For instance, decentralised oracles are used in decentralised prediction markets such as *Augur*[1] and *Gnosis*[2] to support placement of wages on future real-world events. Decentralised oracles also find application in smart securities, derivatives, insurance, Trade Finance, etc.

Typically, on the native web, oracles can be thought of as third-party APIs. However, a critical factor to consider for deployment of oracles on blockchain platforms is the absence of 'a trusted party'. Therefore, it is crucial that decentralised oracles are used to provide data to blockchains, to prevent single point of failure. For example, *TownCrier* [ZCC+16b] provides an authenticated data feed for blockchain platforms by using trusted hardware (Intel's SGX) to retrieve data from *trusted* HTTPS-enabled website and serving same to smart-contracts on the Ethereum blockchain. TownCrier adopts simple majority voting among multiple websites offering the same information, and multiple SGXs. Similarly, *Chainlink*[3] proposes a decentralised network oracle to

[1] https://augur.net/
[2] https://gnosis.io/
[3] https://link.smartcontract.com/whitepaper

solve the oracle problem of blockchain platforms and smart contracts. To prevent single point of failure, Chainlink relies on many nodes to source responses for off-chain data requests.

Our proposed voting scheme can be deployed within decentralised oracle networks to determine 'truth' among competing solutions or answers proposed by oracle nodes as response to data request on a decentralised oracle network. The voting scheme can be deployed in a majority election rule to determine the 'truth' among the available options.

We remark that computational expense (gas in Ethereum) is a key factor that impacts design choices for contracts on the Ethereum blockchain. Code execution cost on Ethereum is standardised using gas, and opcode's gas cost is based on its expected execution time. Gas is the measure of computational complexity of a transaction on Ethereum, i.e. the more the operations, the higher the cost of gas needed for execution.

Naturally, similar to xMcCorry et al. in [MSH17], the voting protocol can be deployed as multiple smart contracts directly on the Ethereum blockchain and rely on nodes the network to ensure the correct execution of the protocol. A single smart contract that handles the election process and the logic of the voting protocol and zero-knowledge verification, and another contract for distribution of zero-knowledge proofs required for correct protocol execution by the parties involved. The second contract is to ensure all participants use the same codes for the creation and verification of the zero-knowledge proofs. The codes for the second contract are to only be run by users locally (not on the network).

The cost of computing reconstructed keys in [MSH17] is the predominant factor for deciding the number of voters the scheme can support. Specifically, because the reconstruction of the keys must be done in a single transaction, and the block's gas limit must not be surpassed by the gas used for the reconstruction. However, the contract can be modified to perform the processing in batches and allow multiple transactions to complete the task. Thereby, supporting a voter size beyond the current 50 voter upper limit. Moreover, the authors explain that a dedicated blockchain will *almost certainly be a requirement* to support a large-scale election.

Alternatively, instead of carrying out all computations on the smart contract, a more efficient approach would require significant off-chain computation as shown in [SGY20]. This approach would involve incorporating the off-chain process in a publicly verifiable and efficient way to reduce on-chain computation on the Ethereum blockchain.

To improve efficiency and minimise gas costs in the execution of the protocol, several cost optimisations will be required. Specifically, it will be important to use the recently introduced Ethereum pre-compiled smart contracts for elliptic curve operations [Rei17] because the cost of gas is lower in pre-compiled smart contracts in comparison to native Ethereum smart contract. s For deployment, the voting protocol can be encapsulated in a smart contract and deployed on the Ethereum blockchain. The smart contract holding the voting protocol logic can be deployed to use the contract address of an ERC-20 token contract as its constructor variable. Holders of the token can then participate in the voting process using transactions sent to the address of the contract smart contract.

The smart contract will require the token cap data for all token holders of a given ERC-20 token, which can be computationally expensive depending on the actual number[4], or associated Merkle proof [SGY20].

4.6 Summary

In this chapter, we described a privacy-preserving delegable blockchain-based voting scheme for general decision-making. We provide a brief discussion of existing works in practice similar to our scheme by highlight key differences (limitations) in comaprison to our proposed scheme. Our solution mainly leverage concepts and building blocks from our treasury system presented in Chapter 3, to support voting on blockchains with private stake information.

The presentation of this chapter further supports the presentations in Chapter 3 in relation to the main research question of this thesis. Specifically, in addition to addressing the thesis research question, it extends the scheme in Chapter 3 to support blockchains

[4]Some tokens have as small as 6 users and others as high as over 3 million holders. See https://etherscan.io/tokens for details

with private stake information.

Specifically, we provide a new multiplicative relation non-interactive zero-knowledge (MR NIZK) proofs to prove the relationship between two unit vector ciphertexts and an encrypted scalar value. This allows a voter to submit a vector encryption that is essentially an encryption of their stake multiplied by a unit vector encryption (same as in the treasury system but already weighted with the voter's stake). Further, we provide formal descriptions of the MR NIZK proofs and the extended polling scheme.

Finally, to enhance utility of decisions reached on the system, we provide metrics that enable system participants evaluate their decisions within the context of collective solutions reached on the blockchain. These metrics are 1) *distance measure* which covers the distance between individual choices and collective choice (system outcome) 2) *consensus measure* which is essentially a weighted aggregation of individual distance measures and 3) *proximity measure* which is an aggregation of individual distance measures across all the proposals voted on in a system period.

Next, in Chapter 5, we provide a security analysis of our proposed systems using traditional security properties in the voting literature. We also provide details of implementations and benchmarks of tests of prototype implementations of the proposed schemes.

Chapter 5

Analysis and Implementation

Following the presentations of the treasury system in Chapter 3 and the privacy-preserving general decision-making system for blockchains in Chapter 4, in this chapter we formally analyse the security of the designed systems. Specifically, we evaluate the security of our proposed system by examining how the system satisfy traditional securities properties expected of blockchain-based and e-voting systems e.g. robustness, ballot secrecy, voter eligibility enforcement, fairness, verifiability, coercion-resistance, [DGS03, KY04, KKW06, Adi06, CCFG16, Adi08, CZZ+19]. Formal proofs of security of the systems can be found in Appendix B. Moreover, we discuss the security of the designed voting (polling) protocol under the UC static corruption model. Thus, we prove that the protocol UC-realises the ideal voting functionality $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$, under the stated static corruption model.

Furthermore, we present information on implementation and performance of the prototypes. The result of the implementation confirms the feasibility and practicality of the proposed systems for use within real-world cryptocurrencies or general blockchain technologies.

The remainder of this chapter is organised as follows. Section 5.1 examines the security of the proposed systems using conventional properties of secure electronic voting systems in the literature. Thereafter, in Section 5.2, we present details of implementation and performance of the complete prototype. Particularly, we elaborate on prototyping information and details of the test network used in the experiments to evaluate the

practicality of the proposed blockchain-based systems. Subsequently, Section 5.3 details the evaluations and benchmark of the prototypes. Specifically, it provides graphical representations of the communication and proof size of the prototypes implementation.

5.1 Security Properties

Below we discuss our scheme within the context of conventional electronic voting security properties established in the literature [Gri03, BY86, KY04, Adi08, Ben06, DGS03]. Specifically, for each listed property, an explanation of how our scheme satisfies the given property is provided. Our analysis implicitly assumes honesty or accurate functioning of voting clients of all participants.

- **Robustness / Reliability:** In our voting scheme, it is unconditionally guaranteed that any individual voter, stakeholder, or participant may fail resulting in no negative effect on the protocol, i.e. there is no single-point of failure. Furthermore, through the use of distributed threshold key generation and decryption, our protocol is not affected when any number of voting committee members $\mathcal{C}$ less than the threshold t fail or become corrupted. The protocol guarantees termination under the given scenarios.

- **Privacy:** This property guarantees that no information about individual ballots is leaked except the final tally result. Our scheme achieves this through the CPA-security of the elliptic curve lifted ElGamal encryption scheme under which individual ballots are encrypted. Moreover, the additively homomorphic property of lifted ElGamal, which is exploited for adding individual ballots to obtain the voting result in our tally algorithm, does not require that ballots are decrypted. Therefore, an adversary cannot learn additional information about the voter's choice by observing the unit vector ballot ciphertext.

- **Eligibility / Authentication:** This property ensures that only qualified parties are authenticated and allowed to vote. Specifically, in our scheme, this property is achieved through a registration stage at the beginning of the system epoch.

Also, we remark that given the decentralised blockchain architecture, every eligible participant is able to participate and vote in the decision-making process by submitting a valid stake-locking registration transaction (that also assigns them a voting power corresponding to their locked-stake).

- **Uniqueness:** Closely related to the authentication property of ensuring only eligible users participate in the decision-making process, the uniqueness property generally ensures that no voter should be allowed to vote more than once. However, within our scheme, we construe uniqueness to guarantee that users only submit a number of valid votes relative to their locked stake. In other words, a user's vote is directly proportional to their voting stake. In our scheme, we guarantee that a user does not gain any advantage by splitting their stake into multiple accounts, each with a smaller fraction of the original stake (Sybil attacks). We remark that although users can submit multiple ballots during the voting stage, only the latest valid ballot is included in the tally process. The motivation for multiple voting is to enable users change their decisions e.g. vote directly after initially delegating their voting power or contrariwise, throughout the voting stage. Furthermore, we prevent ballot copying through the use of encrypted ballots, i.e. a voter (respectively expert) cannot copy other voters' (respectively other experts') ballot without the knowledge of the randomness in the encrypted ballot.

- **Integrity:** Through the use of non-interactive zero-knowledge proofs and specialised blockchain transactions, the scheme guarantees that only properly formatted unit vector ballots are considered in the tally process (and discards of invalid ballots). Thus, guaranteeing the correctness or accuracy of the voting outcome because only validly recorded ballots published on the blockchain, are used in the tally process. The inherent immutability of blockchain transactions prevents the adversary from altering the recorded blockchain ballots without detection by other parties on the network. Moreover, in a worst-case scenario where the majority of the voting committee members are malicious, the integrity of the voting result is guaranteed. However, the privacy of the ballots become compromised in this case.

- **Fairness:** The semantic security of the lifted ElGamal homomorphic encryption scheme and the non-interactive zero-knowledge proofs/arguments of correct unit-vector ballot encryption helps guarantee that no partial sum or intermediate tally results can be revealed before the tally is completed. Therefore, for any voter/expert, their voting decision in not influenced by the recorded ballot (on the blockchain) of how other parties voted, i.e. a party gains no advantage by delaying their voting decision until after others have submitted their ballots.

- **Coercion:** The distributed and decentralised nature of blockchain-based voting makes large-scale coercion difficult to achieve (when compared to conventional voting systems) because of the high technical and monetary cost of efficiently coordinating, enforcing and ensuring that users comply with coerced/forced voting choices. Furthermore, rational honest voters within the system are incentivised to support decisions that improve the overall health of the platform (underlying cryptocurrency) because of the direct implications of bad decisions on their held stake. Therefore, they can submit multiple ballots to falsely 'convince' a coercer of voting in a manner required by the coercer (because only the last valid ballot is considered in the tallying process). However, we note that it is impossible to eliminate coercion risk because a 'willing' voter can supply the coercer with their credentials. Also, a coercer watching over the shoulder of a voter can physically ensure the voter's ballot corresponds to his decision.

- **Voter anonymity / pseudonymity:** Our construction inherits the privacy e.g. anonymity or pseudonymity , provided to users by the underlying blockchain. Specifically, participation in the voting process does not reveal additional user information (except the fact/observation that they submitted a ballot). Therefore, there is no association between identity of voters and how they voted.

- **Verifiability:** Verifiability allows observers/participants to verify that votes have been recorded, tallied and declared correctly. Following the definitions of [KRS10], our scheme satisfies individual, universal and eligibility verifiability. The individual

verifiability property ensures a voter can check that their ballot is successfully 'published' on the underlying blockchain while universal verifiability guarantees that any party can check that the tally corresponds to published ballots on the blockchain. Additionally, eligibility verifiability covers the requirement that any party can verify that all ballots published on the blockchain are from eligible voters. This property is particularly useful in preventing against ballot stuffing. Evidently, our scheme satisfies individual and universal verifiability because a voter can verify their vote is properly recorded on the blockchain and anyone can verify that the final 'encrypted' tally corresponds to the homomorphic addition of all valid ballots recorded on the blockchain and their associated zero-knowledge proofs of validity. Also, any party can verify the tally outcome by combining published decryption shares (accompanied with valid proofs of correct decryption) from the committee members and comparing with the outcome from the committee. Moreover, our scheme achieves eligibility verifiability because the registration transaction can be used to support eligibility checks in verifying that ballots recorded on the blockchain are from eligible participants. We remark that this verifiability relies on the fact that the voting committee is cryptographically distributed.

5.2 Implementation and Performance

Here, we provide details of the implementation for both the treasury system and privacy-preserving decision making system for blockchain governance. Mainly, we produced two independent implementations of the main algorithms and unit vector non-interactive zero-knowledge proofs in Scala and Python. The implementations are based on elliptic curve groups for efficiency. The results from these distinct implementations were compared for consistency and correctness. The Python implementation[1] uses Petlib [Dan] - a Python library that implements a number of privacy enhancing technologies. Moreover, the Scala implementation also includes a full-prototype of the systems built upon Scorex 2.0 [IOH] - a modular blockchain framework for designing blockchain systems. Therefore,

[1] https://github.com/balogunh/TreasuryPrototype

the systems rely on Scorex 2.0 for the underlying blockchain functionalities of storing ordered sequences of transactions and messages. Specifically, the transactions or messages involved are:

- proposal submissions
- voter, expert and committee stake locking
- voting results
- additional data e.g. signatures for results or proofs for verification of results, randomness for committee member selection

Note that to minimise storage space on the blockchain, service messages for the decision-making processes are stored in a dedicated side-chain (whose entries are also anchored to the main chain). We now provide more detailed information about the implementation prototyping.

5.2.1 Prototyping

For the prototype implementations, the voting protocol carries the main computational and storage workload. Therefore, performance testing and evaluations focuses on the voting protocol. The proposed systems were implemented as a fully functional cryptocurrency prototype using the Scorex 2.0 framework to provide the underlying required blockchain functionalities. Scorex 2.0 is a flexible modular framework designed particularly for fast prototyping with a rich set of already implemented functionalities such as asynchronous peer-to-peer network layer, built-in blockchain support with pluggable and extensible consensus module, simple transactions layer, JSON API for accessing the running node, etc.

Socrex 2.0 offers some key benefits for rapid efficient prototyping in comparison to other blockchain frameworks. These include, its flexible modular design, compact functional code written in Scala programming language, full-fledged implementation of a blockchain system built on top of Scorex - TwinsCoin, a hybrid Proof-of-Work (PoW) and

Proof-of-Stake (PoS) cryptocurrency [CDFZ17]. Hence, we adapt and extend Twincoin with the adequate functionalities for the voting protocol. For example, treasury integration requires modification of the existing transactions structure and block validation rules, as well as introduction of new modules for keeping treasury state and prevent transaction forging and other relevant attacks.

To ease code maintenance, we also adopt a modular approach to developing different elements of the voting protocol e.g. distributed key generation. All cryptographic protocols related to the voting procedure were implemented in a separate library. Therefore, these protocols can also be reused within other blockchain systems or within other standalone voting systems. The cryptographic protocol implementation uses BouncyCastle [2] library (ver.1.58) that provides required elliptic curve operations. Additionally, some operations in the finite field were implemented with help of the BigInteger class from the Java Core. Note that sub-protocols of the developed system were implemented exactly as described, without any protocol-level optimizations.

In the pre-voting stage (see Fig.3.9), the implemented registration procedure includes functionalities proposal submission, voters, experts and committee members registration and random selection of the voting committee. Locked stake (security deposits) were implemented through locked unspent transaction output (UTXO) records. All system events are recorded in the blockchain so everyone can easily verify correctness of the operations at each stage of the voting protocol.

The implementation also contains a full-fledged distributed key generation protocol with non-interactive zero-knowledge proofs for correctness of messages. The protocol is resistant to the corruption of up to half of the committee members. During the protocol execution, a list of malicious members is maintained. This is used to prevent corrupted committee members from further participation in further stages of the protocol and allows for punishing them (through stake forfeiture) in the future.

Furthermore, the voting ballot is implemented as a special ballot transaction and it is also accompanied with a non-interactive zero-knowledge proof of correctness (of unit

[2] a collection of Java cryptographic APIs

vector encryption). Invalid ballots are rejected when their verification fails and are never be included in the blockchain. For the post-voting stage separate transactions are used for joint decryption of tally, randomness submission for the next system epoch, punishment for the malicious and disqualified members, incentive rewards for participants, locked stake or deposit refund and distribution of payments for approved/winning proposals.

We remark that most of the operations are automatised and do not require direct user involvement except for the proposal submission and ballots casting operations. A user only needs to configure the node properly and run it to participate in the system. All complex operations such as randomness extraction, distributed key generation, joint tally decryption and payments are implemented as completely automated operations. Also, each stage lasts a fixed number of blocks and comprises its own special transactions. These transactions are included in the blockchain if and only if valid against the current (treasury and general-decision making) system and blockchain state.

5.2.2 Test Network

The test of the developed system prototypes were carried out in a real environment comprising a local network of 12 blockchain full-nodes. The deployment was successfully run for several days spanning dozens of system epochs. A completed cycle consists of 780 blocks (or approximately 4.5 hours treasury cycle) because the underlying blockchain (TwinsCoin) consensus had a block generation time of 10 seconds. The network had 9 voters, 3 experts and 12 candidates with varying amount of cryptocurrency stake. Besides, there are 12 candidates for membership of the voting committee from which 10 were selected. For the tests carried out, the number of proposals varied from 1 to 7.

Furthermore, the tests involved the simulation of multiple unusual or adversarial conditions. For instance, a malicious committee member in the distributed key generation provides incorrect shares, non-participation by some experts or voters in the actual voting process despite registering, refusal to participate in the decryption of tally by some voting committee members, etc. As long as there exist a working honest majority of the voting committee members, the voting outcomes (tally) were always successfully obtained and

rewards were correctly distributed to participants.

5.3 Evaluations

For evaluating performance of the cryptographic protocols a special set of tests were developed as a part of the cryptographic library. Tests were run on the working station with Intel Core i7-6500U CPU @ 2.50GHz, 16GB RAM, running Linux Ubuntu 16.04 64 bit and scalaVersion 2.12.3 . All performance benchmarks are obtained for single-threaded version of the implementation. Providing multi-threaded execution and native code implementation, as well as protocol-level optimizations, allows for further improvement of running time.

We are interested in the computational and communicational complexity of the messages for processing the generation of the shared secret/public key, ballot casting and validation, and tally calculation. Thus, we evaluate the computation time and space requirement of the distributed key generation protocol execution, voters' and experts' ballots processing and tally calculation during one system cycle (epoch).

We benchmarked key generation protocol running time for varying number of voting committee members e.g. from 10 to 100. Note that, higher numbers of committee members might be required to guarantee honest majority on random selection from a large amount of system participants. Moreover, we generated shared public keys for the setting where all committee members are honest, as well as other settings with malicious committee members. With a minority malicious voting committee members, the exact number or ratio of malicious parties does not influence the protocol running time for any honest participant. Fig. 5.1 depicts the running time of the distributed key generation protocol.

Clearly, as seen in fig. 5.1 the overall running time of the protocol increases with increasing number of committee members and ratio of malicious committee members. For example, the DKG protocol execution takes about 0.3 second and 2.4 seconds for fully honest committee sizes of 40 and 100 respectively. Moreover, to complete distributed key

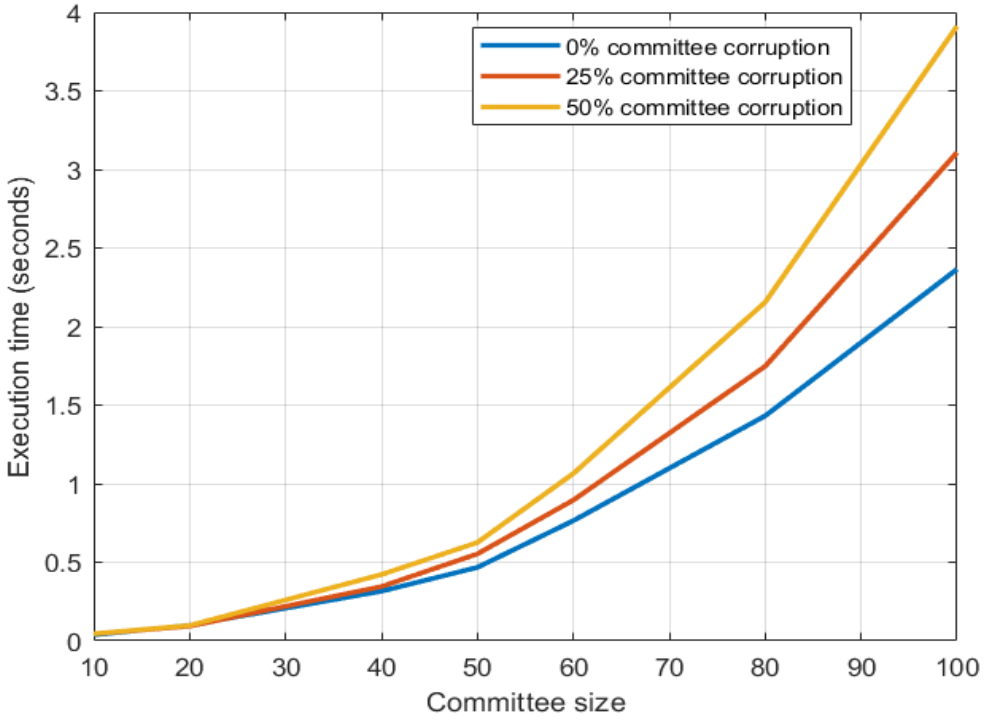

Figure 5.1: DKG protocol execution time depending on the number of committee members

generation protocol execution, it is necessary to transmit data/messages (communication) over the peer-to-peer network. The dominant factor in the execution time is the size of the committee because each member committee member needs to verifiably share their secret with all other members of the committee according to the protocol in fig. 3.6. Latency is a real-world factor that affects the execution time of the DKG protocol shown in Fig. 5.1 if the tests involved geographically distributed committee members. However, this is accommodated by the fact that the protocol is expected to occur over a given time frame where the involved parties post appropriate data (transactions) to the blockchain (semi-synchronous communication). Moreover, a system epoch comprising the election process is designed to take place over an extended period of time, e.g. 30 days.

Expectedly, the communication size depends on the size of the committee and the ratio of malicious committee members. Similar to DKG protocol execution time, the size of the committee and corruption ratio are dominant factors in the overall size of messages required to complete the protocol execution. For example, the traffic size for the DKG is

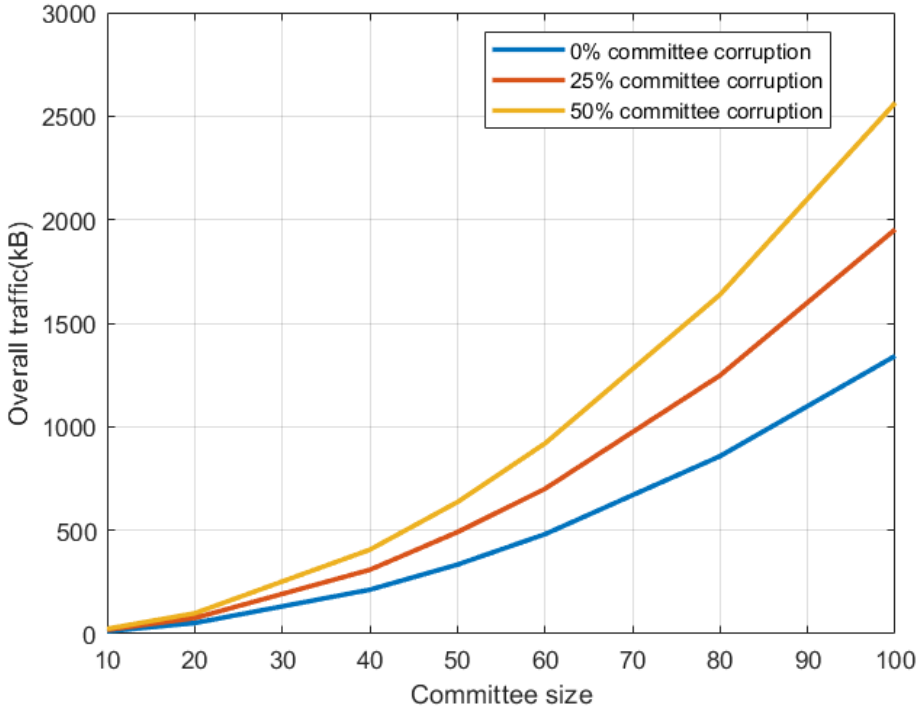

Figure 5.2: Total size of the DKG protocol messages sent over the peer-to-peer network depending on the number of committee members

about 50kB and 800kB for fully honest committee sizes of 20 and 80 respectively. The increase in overall message size for given committee size and varying member corruption ratio is is explained by the fact that additional messages (communication) are needed between honest members to report malicious activities of erring committee members. Further, additional messages are also needed to reconstruct the secret shares of erring members. Fig. 5.2 depicts the communication sizes for varying committee sizes, as well as for different ratios of malicious committee members.

Recall that an adversary controlling 50% of the committee members, can only break confidentiality of voters' ballots, but not the integrity of tally result or individual ballots. For any voter, to generate a single ballot takes less than 1 second in a voting epoch with hundreds of experts. Hence, ballot generation has little influence on the overall performance of the voting protocol.

Meanwhile, *the number/size of experts* is the most important factor in the time it

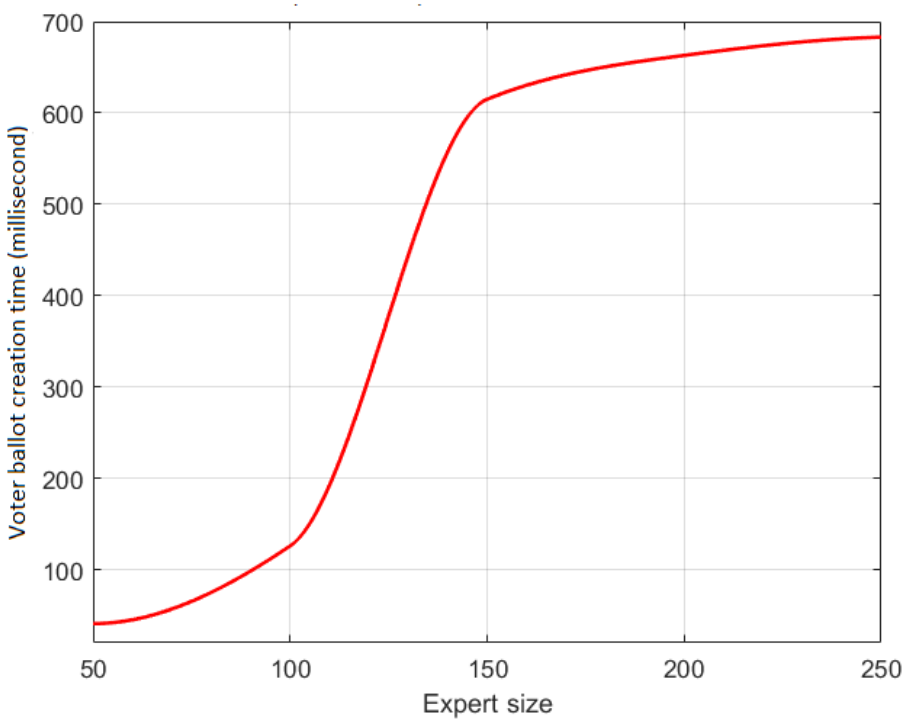

Figure 5.3: Execution time for voter ballot generation with varying number of experts.

takes a voter to generate a ballot. Specifically, since a ballot is encoded as coordinate-wise encryptions of a unit vector, therefore, the length of the vector dominates the time required to create a ballot. As explained in Subsection 3.6.4, the length of the unit-vector, in turn, depends on the number of experts in a given election epoch. Specifically, a voter's unit-vector ballot $(m + 3)$-bits long, where m is the number of experts. However, for an expert, the length of the unit vector ballot is always 3-bits long (because experts vote directly on proposals) and it's creation time is also negligible in the overall protocol performance. Fig 5.3 depicts typical running time for a voter to generate ballot for varying number of experts.

However, unlike ballot generation, the tally process involves considerable computation. Specifically, to obtain tally outcomes, all ballots submitted to the blockchain are collected and their correctness is validated using their accompanying NIZK proofs. Subsequently, the tally is computed using only ballots with valid proofs (that pass the verification checks). The tally involves homomorphic addition of expert ballots, and voters' ballots, and

subsequent joint decryption to obtain tally. Given a fixed, expert size, the computation here is dominated by the number of voters. Specifically, the number of voters directly affect the number of homomorphic additions required for computing the tally encryption while the size of experts already influence the length of the unit vector ballots as earlier discussed.

Fig. 5.4 depicts the average time required to actually compute tally results given different number of voters. As shown in the figure, the time to compute the tally of the voting process is linear in the size of voters. We note that, in case there is malicious behaviour by a committee member in the tally computation, the execution time will increase to accommodate the process on reconstructing the shared secret of the malicious member by other committee members.

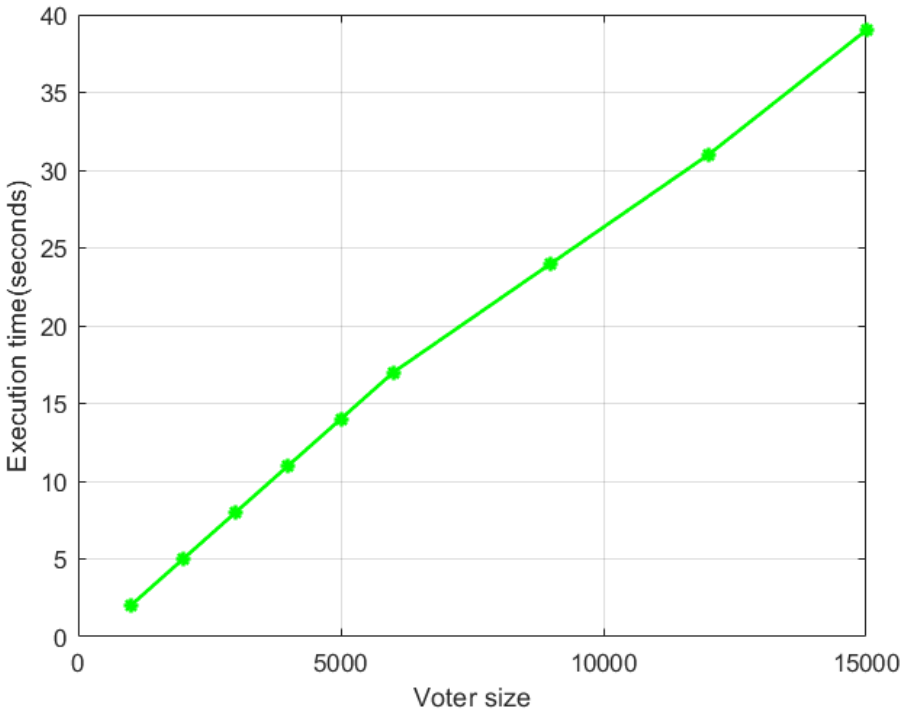

Figure 5.4: Execution time for computing tally outcomes of the entire voting process with varying voter size.

Fig. 5.5 shows the prover's running time, the verifier's running time and the size of the unit vector zero-knowledge proof that has been used in the ballot casting. The time

to create the NIZK proofs is mostly influenced by the size(length) of the unit vector ballot. This length $(m+3)$ is dominated by the number of experts, m in a voting epoch. As seen in fig. 5.5, there is a sharp increase in the unit vector NIZK creation time between unit vector size of 250 and 500. The time increase from about 0.6sec to about 3.4sec is attributed to increased computation for NIZK proof, which is caused by the increase in number of experts (which is the direct implication of increasing the size of the unit vector). Clearly, this negatively impacts the scalability of the system for large values of experts, m. However, in practice the number of experts expected in an election epoch is smaller than 500 and these proofs along with corresponding ballots are generated locally by voters. Further discussion on anticipated sizes in real-world systems is provided in Section 5.4.

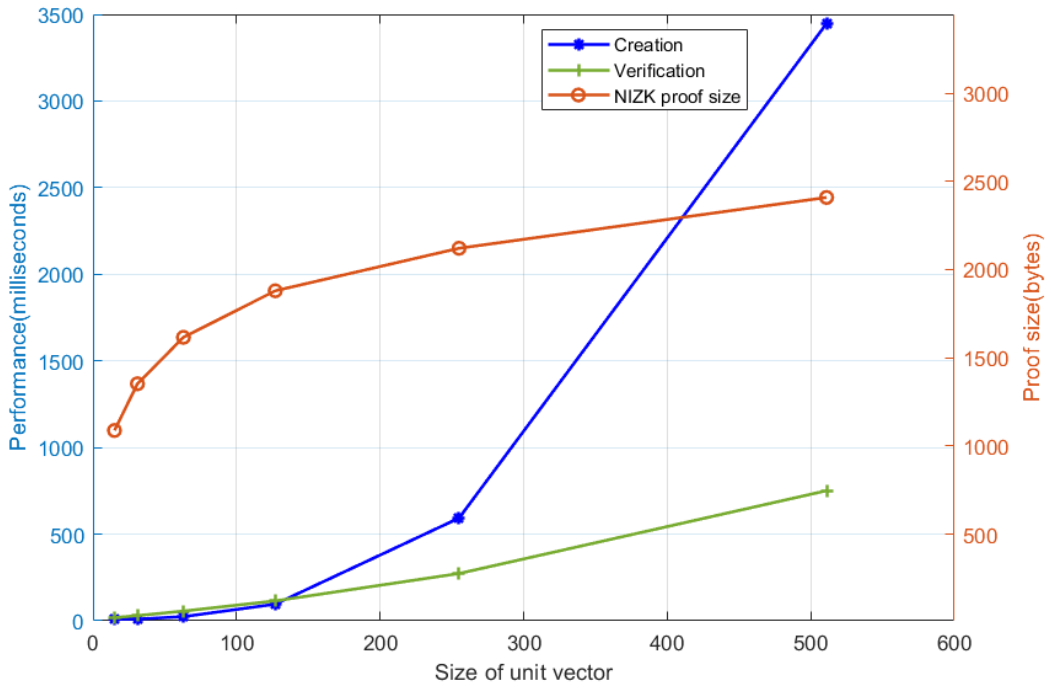

Figure 5.5: The prover's running time, verifier's running time and the size of the unit vector ZK proof.

Recall that the privacy-preserving decision-making system requires additional vector multiplicative relation zero-knowledge proofs. Hence, fig. 5.6 depicts the prover's running time, the verifier's running time and the size of the needed zero-knowledge proof. Similar

to the special honest verifier unit vector zero-knowledge proof in fig. 5.5, the length of the unit vector ballot is the dominant factor in time to create and verify the proof. The proof creation time scales poorly with increase in the size of the the unit vector (number of experts) in an election epoch as shown in fig. 5.5. Moreover, details of anticipated size of the unit vector ballot in a real-world deployment is provided in Section 5.4, to mitigate the scalability issues in the zk proofs.

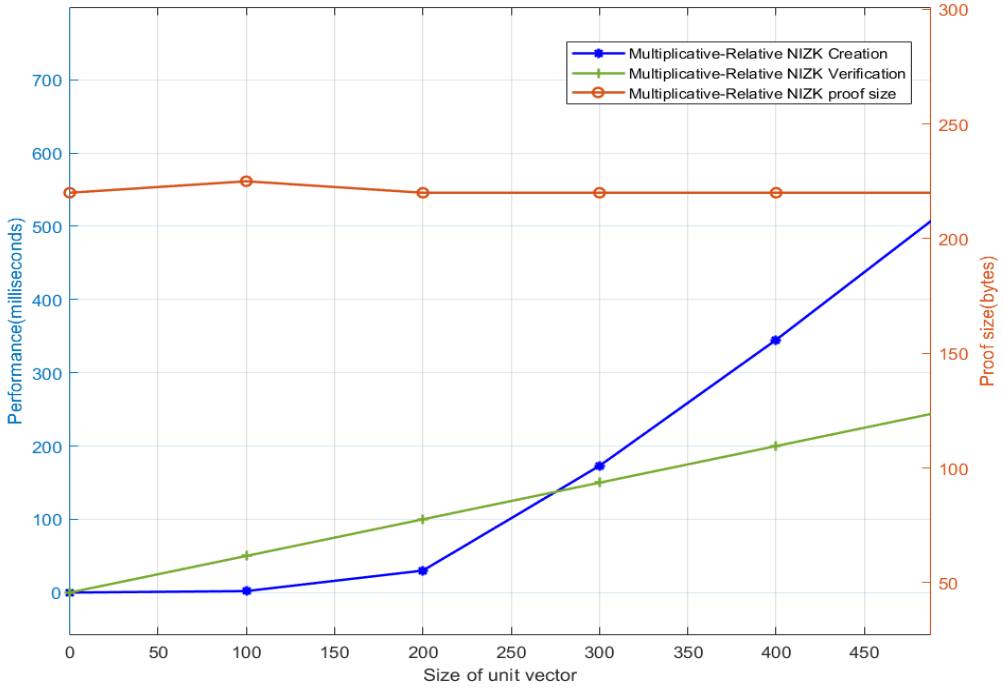

Figure 5.6: The prover's running time, verifier's running time and the size of the vector multiplicative relation ZK proof.

Finally, the overall communication cost for all the voting ballots per project during the entire treasury period and privacy-preserving decision-making is depicted in Fig. 5.7 and Fig. 5.8 respectively. In particular, for a treasury period with 2000 voters and 50 experts, the overall communication is approximately 10MB per project. For a period of the privacy-preserving decision-making system (respectively the treasury system) with 5000 voters and 50 experts the overall communication is approximately 40MB per project

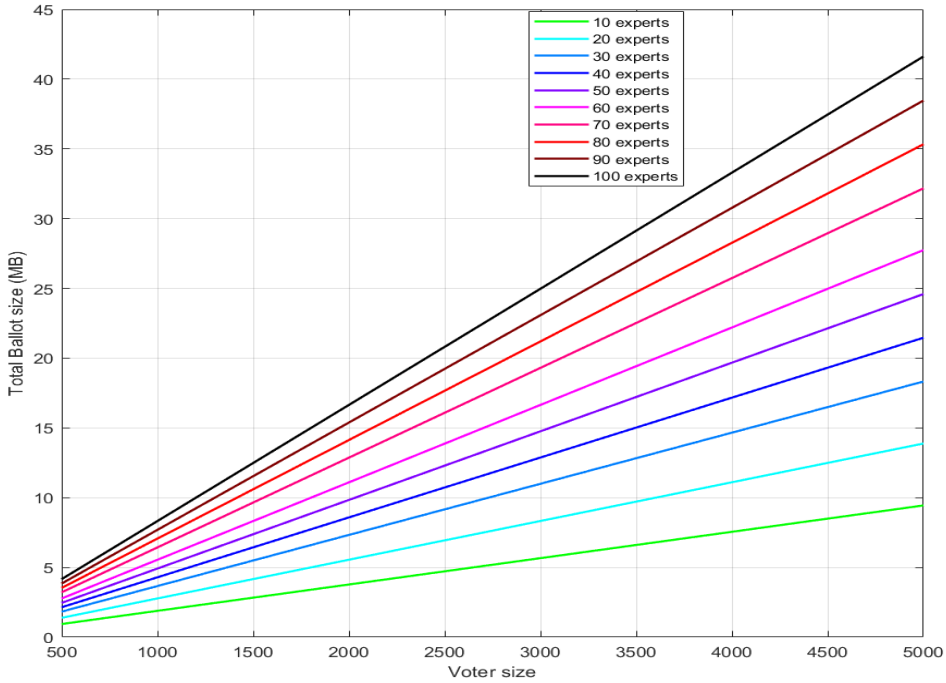

Figure 5.7: The overall communication for all the voting ballots during a treasury period.

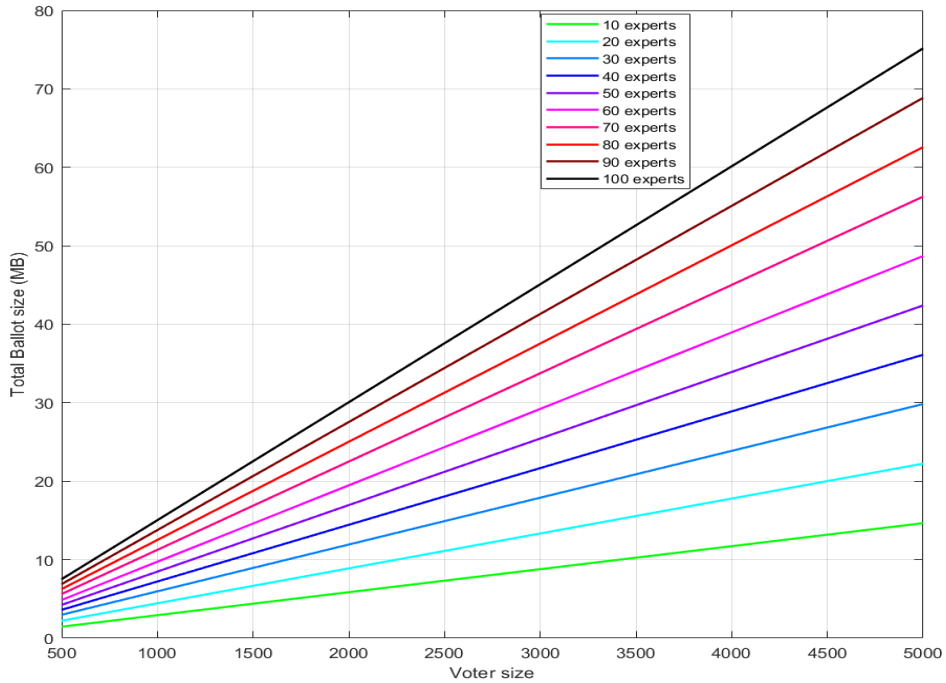

Figure 5.8: The overall communication for all the voting ballots during a period of the privacy-preserving decision-making system.

(respectively 25MB for the treasury system). From fig. 5.7 and fig. 5.8, the communication sizes is dominated by the number of experts (unit vector size) and the size of voters in the voting epoch.

Remark. Note that in practice, the treasury period is long enough, e.g. 30 days (approximately 4320 blocks for Bitcoin), so blockchain space overhead for treasury deployment in the cryptocurrency blockchain is insignificant. Moreover, we consider a sidechain approach [BCD+14, KZ19, GKZ19] for the system implementations as an effective solution. This allows for separation of treasury/decision-making functionalities from the main blockchain consensus, thereby providing a number of advantages e.g. maximising transaction block space for other pertinent transactions. In particular, the system protocols do not influence the main blockchain consensus, therefore, enabling transfer of all implementation complexity to a sidechain. Additionally, storage space on the mainchain is preserved for core clients as a result of the modular construction.

5.4 Discussion

In a real-world blockchain, voters only need to create their ballots and associated proofs and post these to the blockchain. For voters, these processes are done on their local clients and the outputs are posted as appropriate transactions to the blockchain network where it is propagated to other nodes. Mainly, the time between submitting ballot transactions and confirmation that the transaction has been committed by majority of nodes on the blockchain, is a factor to consider in real-world deployment. For example, currently on the Bitcoin blockchain, transaction propagation time to about 90% of all network nodes is about 15 seconds[3] [NAH16].

Additionally, from the evaluations it is clear that the size of the unit vector is a very dominant factor in the performance of the protocols. However, since the size of a unit vector ballot is exactly three more than the number of experts, i.e. $|\textit{unit vector ballot}| = |\textit{experts} + 3|$, some of the performance bottleneck can be mitigated. Specifically, this requires that the number of experts is set to a reasonably practical number.

[3] https://www.dsn.kastel.kit.edu/bitcoin/

One way of doing this, is through classification of experts based on the topics of proposals such that no individual can participate as an expert across all the areas being voted on. Moreover, an alternative approach would involve ranking of available experts based on a so-called *reputation or prestige* [KSG03, WS06, Szt15, PK15, dPLC17, HWC+21, KSA+21] and allowing a subset of the top-ranked experts to participate in the voting process. Essentially, reputation is a measure of quality of contributions (or honesty) of the expert to select activities e.g., consensus building, on the blockchain.

Now, we provide discussion on potential numbers of a deployment of our proposed systems in a real-world cryptocurrency. This anticipated size is based on numbers from real-world deployments of cryptocurrency treasury system [4]. Given the recent summary from the Cardano Project Catalyst Fund 4[5] with about 216,749 ballots across all 267 proposals, spanning categories such as dapp integrations, distributed decision-making, developer ecosystem, proposer outreach, etc. and 56 winning proposals[6]. This implies an average of about 800 individual ballots per proposal. Note that the total number of ballots cast for a proposal does not depend on the amount of stake involved. Rather, each ballot is weighted by the stake of its 'caster' in the tally process.

The anticipated communicational overhead of a proposal for a treasury system (respectively privacy-preserving general decision-making system) with 1000 ballots and 100 experts is estimated at about 8.3MB (respectively, 15.3MB). The increase in size in the privacy-preserving system is attributed to the additional multiplicative relation zero-knowledge proof required in the ballot casting process. Moreover, in a given system epoch, with an expert size of 50 and $10,000$ voters , the anticipated communicational overhead for a single proposal is about 74MB .

[4] https://bit.ly/3rf4p4v
[5] https://bit.ly/3xMiWXT
[6] https://bit.ly/2Uo6hvJ

Chapter 6

Conclusion

The goal of this thesis has been to construct provably-secure community-inclusive cryptographic protocols that exploits the decentralised nature of blockchains and optimises community intelligence through appropriate decision-making systems. The work presented in this thesis explored sustainable development and inclusive maintenance of blockchains.

The importance and proliferation of blockchain technology has been well documented, both in the literature and in practice. The decentralised architecture of blockchain technologies that enable (sometime) mutually distrustful parties to engage in a computation tasks without the need for a 'trusted third party' or single point of failure has enabled the deployment and adoption of blockchains across multiple sectors such as finance, healthcare, futures and prediction markets, retail, etc.

However, this consensus layer decentralisation or decentralised ethos of blockchains is often not reflected in the development and management of blockchains. Typically, decisions on development and maintenance of these platforms serves as a source of centralisation in the system with 'powerful minority' decision-making. Moreover, some of the prevalent practices have been described as *benevolent dictatorship* [AMM18]. Consequently, there has been a number of cryptocurrency platforms such as Dash with DGBB, to provide treasury system support for blockchain development. The goal is to democratise the development process of the blockchains, by providing funding support

for community members (or any interested parties) with proposals that improve the blockchain in particular or the general cryptocurrency ecosystem.

As demonstrated in this thesis, we explored existing solutions and literature and provide an affirmative answer to the question - *is it possible to have provably secure community-inclusive blockchain-based self-sustenance mechanisms that will ensure long-term sustainability of cryptocurrencies?* Specifically, we identified and explored the role of funding (and funding source) in the overall management of blockchains. We highlighted the various funding sources and their suitability towards long-term cryptocurrency development funding. In our proposed treasury, system we adopt a hybrid funding that guarantees the availability of funds for cryptocurrency development funding (even with decreasing miner's reward) and does suffer from the deficiencies associated with a funding, for example, based only on donations or patron organisations.

To optimise community-inclusion in contributions to blockchain development and allocation of blockchain development funds, for the first time, we proposed and developed a UC-secure blockchain-based voting scheme (for treasury systems) that supports liquid democracy. It is an incentivised voting scheme, and supports private ballots and is robust to adversarial attacks from the corruption of less 50% of the voting committee members. We guarantee the integrity of election outcome in the rare scenario where all committee members are corrupt using the zero-knowledge proofs which are publicly posted to the blockchain.

Given the increased interest in decentralised governance in blockchains, we extend the applicability of the delegable voting scheme at the core of our treasury system and proposed a privacy-preserving system to support general decision-making on blockchains with private stake information. Besides, to enhance community acceptance and increased participation in the decision-making process, we provide a method for evaluating 'consensus' on decisions reached within the system. The analysis utilises a set of distance measure metrics.

Moreover, for improved efficiency, we propose a novel honest-verifier zero-knowledge proof for unit vector ballot encryption with logarithmic size communication. Whereas,

similar homomorphic-tallying based cryptographic schemes such as Helios [Adi08] and its variants, the proof size is linear in the size of the ballot.

We provide implementations of the the algorithms and full-prototype of the proposed systems (treasury and privacy-preserving general decision-making) and benchmarks from tests on the feasibility of the proposed schemes for use in real-world blockchains. The prototype leverage the Scorex framework which utilises TwinsCoin consensus for the underlying blockchain. Results show promise for adoption in real-world systems.

Recall that the voting schemes requires locking (freezing) the deposits (stakes) of entities who participate in an election epoch. This mainly serves as a mechanism to incentivise correct behaviour from the different participants. For example, locked deposits of election committee members and experts who behave maliciously are confiscated. Moreover, for other voters, this provide an incentive to vote in support of proposals that are healthy to the overall growth of the blockchain. This is because voting for proposals that for example negatively affect the exchange rate of a cryptocurrency directly affect the value of their locked stake. Further, without the coin locking, it is trivial for an adversary to collect sufficient coins for a single snapshot of the system epoch for the purpose of attacking the scheme.

Despite the above incentivisation provided by locking stake, it is true that this affects the usability of stakeholders. For example, owners of locked stake cannot spend their coins for other purposes e.g., purchasing a good online, while it is locked. For general deployment of the voting schemes proposed in this thesis outside of blockchains (cryptocurrencies), malicious activities by participants will still be detected. However, it is left to be decided what the incentive (reward/punishment) mechanisms will be to encourage honest behaviour by the entities involved in the execution of the protocol.

In our schemes, to prevent Sybil attacks, we assume a flat model of stake distribution, i.e. every stakeholder owns exactly 1 coin, and voting power is directly proportional to the amount of stake owned. Indirectly, this can lead to a situation where blockchain development decisions are mostly made by the wealthy (plutocracy). Given the pseudonymity/anonymity provided by blockchains and the separation from

real-world identities of stakeholders, it is difficult to securely distribute voting powers (one-man one-vote) based on real-world identities. However, there are potential measures to mitigate excessive influence or dominance by the wealthy in the blockchain schemes. An example of this measure, is to leverage a metric that captures the contributions of stakeholders based on their activities on the blockchain e.g. *reputation*, and allow users that meet a certain threshold participate in the election process (without the requirement to have sufficient funds). This is particularly useful for allowing community members with technical know-how to serve as experts in the decision-making system even if they do not possess the required fund. However, it is also important to achieve a balance and prevent scenarios where it is possible to compromise the underlying blockchains with little to no resources in the system, as an attack against other stakeholders with major stake.

This requires careful design and game-theoretic analysis to ensure secure implementation or adoption and protection against new attack vectors that make it easy to compromise the long-term value and overall development of the underlying blockchain. This is an interesting area of future work considering new results in decentralised identity management platforms such as CANDID [MMZ+20] that offers Sybil-resistance.

Overall, the work presented in this thesis provides insights into new research directions to further developments in blockchain research and beyond.

6.1 Future Directions

In this thesis, we investigated funding sources and decision-making on utilisation of funds in blockchains. Thereafter, we initiated the study of blockchain treasury systems for the community to collaboratively collect and distribute funds, for supporting the system in a decentralised manner. Thereby, eliminating single point of failure or unhealthy centralisation of power. The proposed treasury system is currently adopted in practical deployment in real world cryptocurrencies. In particular, the treasury model is in the Voltaire release [Car18b] of Cardano [Car18a]. Horizen (formerly ZenCash) also implements DAO Treasury Protocol-level Voting System [Hor18] based on our scheme.

We note that, the voting scheme in the privacy-preserving general blockchain decision-making and treasury system can further be enhanced with game-theoretic approaches to optimise individual contributions. Thus, enabling better intelligent collaborative decision-making. However, the current voting schemes do not provide coercion resistance or receipt freeness because voters can prove how they voted to a coercer. The schemes can be improved to achieve receipt-freeness or coercion resistance by adopting the well-known *cast or validate* process of schemes such as Helios. Although, this also does not address the problem of a malicious client edited to output ballot randomness to users.

We remark that the current voting scheme in our proposed blockchain systems of Chapter 3 and Chapter 4 assumes correct/honest operation of the clients of all participants in the system for its security. Additionally, privacy of our schemes can be further enhanced by providing a ranked output of winning proposals without revealing the tally numbers.

Moreover, the treasury and decision-making systems in Chapter 3 and Chapter 4 respectively, can be improved by building on recent advancements in practical Fully-Homomorphic Encryption (FHE) e.g. SEAL [SEA19]. Specifically, the efficiency of the schemes can be significantly improved by exploiting the Single-Instruction Multiple-Data (SIMD) [SV14] operations for ballot creation, as well as the cyclic rotation capability on encrypted vectors in the tally computation process.

Another interesting potential application of our blockchain-based voting system is in efficient decentralised oracle network for blockchain platforms. Finally, it will be interesting to conduct further research, especially as it relates to usability of the proposed system when it is finally in production in real-world cryptocurrencies.

Bibliography

[Ad] Adhocracy. Adhocracy official website. `https://adhocracy.de`. Online; date last accessed: 2017-10-21. [Cited on page 125]

[Th16] The Veritas Team. A review of the dash governance system. `https://files.zotero.net/5834821312/Dash-Governance-System.pdf`, October 31 2016. Online; date last accessed: 2017-10-21. [Cited on page 25]

[Adi06] Ben Adida. *Advances in cryptographic voting systems*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2006. [Cited on pages 20 and 147]

[Adi08] Ben Adida. Helios: Web-based open-audit voting. In Paul C. van Oorschot, editor, *Proceedings of the 17th USENIX Security Symposium, July 28-August 1, 2008, San Jose, CA, USA*, pages 335–348. USENIX Association, 2008. [Cited on pages 57, 58, 59, 61, 112, 134, 147, 148, and 168]

[AG07] Chris Ansell and Alison Gash. Collaborative governance in theory and practice. *Journal of Public Administration Research and Theory*, 18(4):543–571, Nov 2007. [Cited on page 133]

[Ago18] Agora. Bringing our voting systems into the 21st century, version 0.2. `https://static1.squarespace.com/static/5b0be2f4e2ccd12e7e8a9be9/t/5b45ff930e2e72b2215df3d9/1531314069537/Agora_Whitepaper.pdf`, 2018. [Cited on pages 71 and 125]

[AKW19] Yousif Abuidris, Rajesh Kumar, and Wang Wenyong. A survey of blockchain based on e-voting systems. In *Proceedings of the 2019 2nd International Conference on Blockchain Technology and Applications*, pages 99–104, 2019. [Cited on page 63]

[AM16] Syed Taha Ali and Judy Murray. An overview of end-to-end verifiable voting systems. *CoRR*, abs/1605.08554, 2016. [Cited on page 58]

[AmGMA20] Ali Mansour Al-madani, Ashok T. Gaikwad, Vivek Mahale, and Zeyad A.T. Ahmed. Decentralized e-voting system based on smart contract by using

blockchain technology. In *2020 International Conference on Smart Innovations in Design, Environment, Management, Planning and Computing (ICSIDEMPC)*, pages 176–180, 2020. [Cited on page 63]

[AMM18] Sarah Azouvi, Mary Maller, and Sarah Meiklejohn. Egalitarian society or benevolent dictatorship: The state of cryptocurrency governance. In *Financial Cryptography and Data Security - FC 2018 International Workshops, BITCOIN, VOTING, and WTSC, Nieuwpoort, Curaçao, March 2, 2018, Revised Selected Papers*, pages 127–143. Springer, 2018. [Cited on pages 3, 6, 69, and 166]

[AO13] Manuel Araoz and Esteban Ordano. Proof of existence. `https://poex.io`, 2013. [Cited on page 2]

[AR06] Ben Adida and Ronald L. Rivest. Scratch & vote: self-contained paper-based cryptographic voting. In Ari Juels and Marianne Winslett, editors, *Proceedings of the 2006 ACM Workshop on Privacy in the Electronic Society, WPES 2006, Alexandria, VA, USA, October 30, 2006*, pages 29–40. ACM, 2006. [Cited on page 59]

[Ara] Aragon. unstoppable organizations. create value without borders or intermediaries. `https://aragon.org`. [Cited on pages 28 and 30]

[ASB+17] Mustafa Al-Bassam, Alberto Sonnino, Shehar Bano, Dave Hrycyszyn, and George Danezis. Chainspace: A sharded smart contracts platform. *CoRR*, 2017. [Cited on page 2]

[ASB+18] Mustafa Al-Bassam, Alberto Sonnino, Shehar Bano, Dave Hrycyszyn, and George Danezis. Chainspace: A sharded smart contracts platform. In *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society, 2018. [Cited on page 24]

[AVP+18] Muhammad Salek Ali, Massimo Vecchio, Miguel Pincheira, Koustabh Dolui, Fabio Antonelli, and Mubashir Husain Rehmani. Applications of blockchains in the internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 21(2):1676–1717, 2018. [Cited on page 6]

[Bas] Basic Attention Token. introducing blockchain-based digital advertising. `https://basicattentiontoken.org`. [Cited on pages 28 and 30]

[BBCV09] Paolo Boldi, Francesco Bonchi, Carlos Castillo, and Sebastiano Vigna. Voting in social networks. In David Wai-Lok Cheung, Il-Yeol Song, Wesley W. Chu, Xiaohua Hu, and Jimmy J. Lin, editors, *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*, pages 777–786. ACM, 2009. [Cited on page 124]

[BBE+13] Josh Benaloh, Michael D. Byrne, Bryce Eakin, Philip T. Kortum, Neal McBurnett, Olivier Pereira, Philip B. Stark, Dan S. Wallach, Gail Fisher,

Julian Montoya, Michelle Parker, and Michael Winn. Star-vote: A secure, transparent, auditable, and reliable voting system. In *2013 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, EVT/WOTE '13, Washington, D.C., USA, August 12-13, 2013*. USENIX Association, 2013. [Cited on pages 59 and 61]

[BCD+14] Adam Back, Matt Corallo, Luke Dashjr, Mark Friedenbach, Gregory Maxwell, Andrew Miller, Andrew Poelstra, Jorge Timon, and Pieter Wuille. Enabling blockchain innovations with pegged sidechains. `https://blockstream.com/sidechains.pdf`, 2014. [Cited on pages 2, 24, and 162]

[BCG+14] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014*, pages 459–474. IEEE Computer Society, 2014. [Cited on pages 5, 24, 25, 123, and 124]

[BCP+16] David Bernhard, Véronique Cortier, Olivier Pereira, Ben Smyth, and Bogdan Warinschi. Adapting helios for provable ballot privacy. *IACR Cryptol. ePrint Arch.*, 2016:756, 2016. [Cited on page 59]

[Ben87] Josh Daniel Cohen Benaloh. *Verifiable Secret-Ballot Elections*. PhD thesis, Yale University, USA, September 1987. [Cited on pages 54 and 56]

[Ben06] Josh Benaloh. Simple verifiable elections. In Dan S. Wallach and Ronald L. Rivest, editors, *2006 USENIX/ACCURATE Electronic Voting Technology Workshop, EVT'06, Vancouver, BC, Canada, August 1, 2006*. USENIX Association, 2006. [Cited on pages 58, 60, and 148]

[BF78] Steven J. Brams and Peter C. Fishburn. Approval voting. *American Political Science Review*, 72(3):831–847, 1978. [Cited on page 72]

[BFL+12] Jonathan Ben-Nun, Niko Fahri, Morgan Llewellyn, Ben Riva, Alon Rosen, Amnon Ta-Shma, and Douglas Wikström. A new implementation of a dual (paper and cryptographic) voting system. In Manuel J. Kripp, Melanie Volkamer, and Rüdiger Grimm, editors, *5th International Conference on Electronic Voting 2012, (EVOTE 2012), Co-organized by the Council of Europe, Gesellschaft für Informatik and E-Voting.CC, July 11-14, 2012, Castle Hofen, Bregenz, Austria*, volume P-205 of *LNI*, pages 315–329. GI, 2012. [Cited on page 59]

[BFP+01] Olivier Baudron, Pierre-Alain Fouque, David Pointcheval, Jacques Stern, and Guillaume Poupard. Practical multi-candidate election system. In Ajay D. Kshemkalyani and Nir Shavit, editors, *Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing, PODC 2001, Newport, Rhode Island, USA, August 26-29, 2001*, pages 274–283. ACM, 2001. [Cited on page 57]

[BFTK20] Matthias BAUDLET, Doudou FALL, Yuzo TAENAKA, and Youki KADOBAYASHI. The best of both worlds: A new composite framework leveraging pos and pow for blockchain security and governance. In *2020 2nd Conference on Blockchain Research Applications for Innovative Networks and Services (BRAINS)*, pages 17–24, 2020. [Cited on page 65]

[BG02] Dan Boneh and Philippe Golle. Almost entirely correct mixing with applications to voting. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, pages 68–77, New York, NY, USA, 2002. Association for Computing Machinery. [Cited on page 55]

[Bita] Bitcoin Magazine. What is an ico? `https://bitcoinmagazine.com/guides/what-ico`. [Cited on pages 28 and 30]

[Bitb] BitcoinCore. Announcing the bitcoin core sponsorship programme. `https://bitcoincore.org/en/2016/04/04/announcing_sponsorship_programme/`. [Cited on page 31]

[Bit17] BitcoinCore. Bitcoin core sponsorship programme. `https://bitcoincore.org/en/about/sponsorship/programme`, 2017. [Cited on pages 28 and 31]

[BLMR14] Iddo Bentov, Charles Lee, Alex Mizrahi, and Meni Rosenfeld. Proof of activity: Extending bitcoin's proof of work via proof of stake [extended abstract]y. *SIGMETRICS Perform. Eval. Rev.*, 42(3):34–37, dec 2014. [Cited on page 2]

[Blo] Blockchain Library. Online access: New digital resources. `https://blockchainlibrary.org/`. [Cited on page 1]

[BM03] Mike Burmester and Emmanouil Magkos. Towards secure and practical e-elections in the new era. In Dimitris Gritzalis, editor, *Secure Electronic Voting*, volume 7 of *Advances in Information Security*, pages 63–76. Springer, 2003. [Cited on pages 54, 56, 57, and 58]

[BM10] Soheil Boroushaki and Jacek Malczewski. Measuring consensus for collaborative decision-making: A gis-based approach. *Computers, Environment and Urban Systems*, 34(4):322 – 332, 2010. Geospatial Cyberinfrastructure. [Cited on pages 134, 136, and 141]

[BMC+15] Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A. Kroll, and Edward W. Felten. Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21*, pages 104–121, 2015. [Cited on pages 1 and 26]

[BMSS17] Stefano Bistarelli, Marco Mantilacci, Paolo Santancini, and Francesco Santini. An end-to-end voting-system based on bitcoin. In *Proceedings of the Symposium on Applied Computing*, SAC '17, pages 1836–1841, New York, NY, USA, 2017. ACM. [Cited on page 125]

[BMTZ17] Christian Badertscher, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. Bitcoin as a transaction ledger: A composable treatment. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 324–356. Springer, 2017. [Cited on pages 86, 106, and 111]

[BNPW18] Jonah Brown-Cohen, Arvind Narayanan, Christos-Alexandros Psomas, and S. Matthew Weinberg. Formal barriers to longest-chain proof-of-stake protocols. *CoRR*, abs/1809.06528, 2018. [Cited on page 29]

[BPW12] David Bernhard, Olivier Pereira, and Bogdan Warinschi. How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *Lecture Notes in Computer Science*, pages 626–643. Springer, 2012. [Cited on page 59]

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, pages 62–73, New York, NY, USA, 1993. ACM. [Cited on pages 22 and 114]

[BRR+15] Josh Benaloh, Ronald L. Rivest, Peter Y. A. Ryan, Philip B. Stark, Vanessa Teague, and Poorvi L. Vora. End-to-end verifiability. *CoRR*, abs/1504.03778, 2015. [Cited on pages 58 and 59]

[BS15] Alireza Beikverdi and JooSeok Song. Trend of centralization in bitcoin's distributed network. In *2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pages 1–6. IEEE, 2015. [Cited on page 6]

[BT94] Josh Cohen Benaloh and Dwight Tuinstra. Receipt-free secret-ballot elections (extended abstract). In Frank Thomson Leighton and Michael T. Goodrich, editors, *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 544–553. ACM, 1994. [Cited on page 57]

[BU13] F. Bohl and D. Unruh. Symbolic universal composability. In *2013 IEEE 26th Computer Security Foundations Symposium*, pages 257–271, 2013. [Cited on page 22]

[But] Vitalik Buterin. Notes on blockchain governance. `https://vitalik.ca/general/2017/12/17/voting.html`. [Cited on pages 6 and 25]

[BY86] Josh Cohen Benaloh and Moti Yung. Distributing the power of a government to enhance the privacy of voters (extended abstract). In Joseph Y.

Halpern, editor, *Proceedings of the Fifth Annual ACM Symposium on Principles of Distributed Computing, Calgary, Alberta, Canada, August 11-13, 1986*, pages 52–62. ACM, 1986. [Cited on page 148]

[Can00] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. *IACR Cryptology ePrint Archive*, 2000:67, 2000. [Cited on pages 9, 11, 14, 22, and 76]

[Car18a] Cardano. Cardano monetary policy. treasury and fees. `https://cardanodocs.com/cardano/monetary-policy/`, 2018. [Cited on pages 27 and 169]

[Car18b] Cardano. Cardano roadmap. voltaire. `https://cardanoroadmap.com`, 2018. [Cited on pages 5 and 169]

[CC97] Lorrie Faith Cranor and Ron Cytron. Sensus: A security-conscious electronic polling system for the internet. In *30th Annual Hawaii International Conference on System Sciences (HICSS-30), 7-10 January 1997, Maui, Hawaii, USA*, pages 561–570. IEEE Computer Society, 1997. [Cited on page 55]

[CCC+08] David Chaum, Richard Carback, Jeremy Clark, Aleksander Essex, Stefan Popoveniuc, Ronald L. Rivest, Peter Y. A. Ryan, Emily Shen, and Alan T. Sherman. Scantegrity II: end-to-end verifiability for optical scan election systems using invisible ink confirmation codes. In David L. Dill and Tadayoshi Kohno, editors, *2008 USENIX/ACCURATE Electronic Voting Workshop, EVT 2008, July 28-29, 2008, San Jose, CA, USA, Proceedings*. USENIX Association, 2008. [Cited on page 59]

[CCFG16] Pyrros Chaidos, Véronique Cortier, Georg Fuchsbauer, and David Galindo. Beleniosrf: A non-interactive receipt-free electronic voting scheme. In *CCS '16*, pages 1614–1625. ACM, 2016. [Cited on page 147]

[CCL15] Ran Canetti, Asaf Cohen, and Yehuda Lindell. A simpler variant of universally composable security for standard multiparty computation. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 3–22. Springer, 2015. [Cited on page 23]

[CCM08] Michael R. Clarkson, Stephen Chong, and Andrew C. Myers. Civitas: Toward a secure voting system. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 354–368, 2008. [Cited on pages 60, 61, and 62]

[CD17] Gertrude Chavez-Dreyfuss. Blockchain token sale nets $25 million in under 15 minutes, May 2017. [Cited on page 30]

[CDE+16] Kyle Croman, Christian Decker, Ittay Eyal, Adem Efe Gencer, Ari Juels, Ahmed E. Kosba, Andrew Miller, Prateek Saxena, Elaine Shi, Emin Gün Sirer, Dawn Song, and Roger Wattenhofer. On scaling decentralized

blockchains - (A position paper). In Jeremy Clark, Sarah Meiklejohn, Peter Y. A. Ryan, Dan S. Wallach, Michael Brenner, and Kurt Rohloff, editors, *Financial Cryptography and Data Security - FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, February 26, 2016, Revised Selected Papers*, volume 9604 of *Lecture Notes in Computer Science*, pages 106–125. Springer, 2016. [Cited on page 2]

[CDFZ17] Alexander Chepurnoy, Tuyet Duong, Lei Fan, and Hong-Sheng Zhou. Twinscoin: A cryptocurrency via proof-of-work and proof-of-stake. *IACR Cryptology ePrint Archive*, 2017:232, 2017. [Cited on pages 96 and 153]

[CDP19] Fran Casino, Thomas K. Dasaklis, and Constantinos Patsakis. A systematic literature review of blockchain-based applications: Current status, classification and open issues. *Telematics and Informatics*, 36:55 – 81, 2019. [Cited on pages 1 and 64]

[CDPW07] Ran Canetti, Yevgeniy Dodis, Rafael Pass, and Shabsi Walfish. Universally composable security with global setup. In Salil P. Vadhan, editor, *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, volume 4392 of *Lecture Notes in Computer Science*, pages 61–85. Springer, 2007. [Cited on page 23]

[CEC+08] David Chaum, Aleksander Essex, Richard Carback, Jeremy Clark, Stefan Popoveniuc, Alan T. Sherman, and Poorvi L. Vora. Scantegrity: End-to-end voter-verifiable optical-scan voting. *IEEE Secur. Priv.*, 6(3):40–46, 2008. [Cited on pages 59 and 61]

[CELR18] Mauro Conti, Sandeep Kumar E, Chhagan Lal, and Sushmita Ruj. A survey on security and privacy issues of bitcoin. *IEEE Commun. Surv. Tutorials*, 20(4):3416–3452, 2018. [Cited on page 123]

[CF85] Josh D. Cohen and Michael J. Fischer. A robust and verifiable cryptographically secure election scheme (extended abstract). In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 372–382. IEEE Computer Society, 1985. [Cited on page 56]

[CG17] Zoé Christoff and Davide Grossi. Binary voting with delegable proxy: An analysis of liquid democracy. In Jérôme Lang, editor, *Proceedings Sixteenth Conference on Theoretical Aspects of Rationality and Knowledge, TARK 2017, Liverpool, UK, 24-26 July 2017*, volume 251 of *EPTCS*, pages 134–150, 2017. [Cited on page 38]

[CGG19] Véronique Cortier, Pierrick Gaudry, and Stéphane Glondu. Belenios: A simple private and verifiable electronic voting system. In Joshua D. Guttman, Carl E. Landwehr, José Meseguer, and Dusko Pavlovic, editors, *Foundations of Security, Protocols, and Equational Reasoning - Essays Dedicated to Catherine A. Meadows*, volume 11565 of *Lecture Notes in Computer Science*, pages 214–238. Springer, 2019. [Cited on page 61]

[CGGI13] Véronique Cortier, David Galindo, Stéphane Glondu, and Malika Izabachène. Distributed elgamal à la pedersen: Application to helios. In Ahmad-Reza Sadeghi and Sara Foresti, editors, *Proceedings of the 12th annual ACM Workshop on Privacy in the Electronic Society, WPES 2013, Berlin, Germany, November 4, 2013*, pages 131–142. ACM, 2013. [Cited on page 61]

[CGGI14] Véronique Cortier, David Galindo, Stéphane Glondu, and Malika Izabachène. Election verifiability for helios under weaker trust assumptions. In Miroslaw Kutylowski and Jaideep Vaidya, editors, *Computer Security - ESORICS 2014 - 19th European Symposium on Research in Computer Security, Wroclaw, Poland, September 7-11, 2014. Proceedings, Part II*, volume 8713 of *Lecture Notes in Computer Science*, pages 327–344. Springer, 2014. [Cited on page 61]

[CGK+16] VÃ©ronique Cortier, David Galindo, Ralf KÃ¼sters, Johannes MÃ¼ller, and Tomasz Truderung. Sok: Verifiability notions for e-voting protocols. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 779–798, 2016. [Cited on page 58]

[CGS97] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. In Walter Fumy, editor, *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding*, volume 1233 of *Lecture Notes in Computer Science*, pages 103–118. Springer, 1997. [Cited on pages 55, 56, 57, and 61]

[Cha81] David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–88, 1981. [Cited on page 55]

[Cha82] David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *Advances in Cryptology: Proceedings of CRYPTO '82, Santa Barbara, California, USA, August 23-25, 1982*, pages 199–203. Plenum Press, New York, 1982. [Cited on pages 54 and 55]

[Cha88] David Chaum. Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA. In Christoph G. Günther, editor, *Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings*, volume 330 of *Lecture Notes in Computer Science*, pages 177–182. Springer, 1988. [Cited on page 54]

[Cha04] David Chaum. Secret-ballot receipts: True voter-verifiable elections. *IEEE Secur. Priv.*, 2(1):38–47, 2004. [Cited on pages 56, 58, 60, and 61]

[CHSW19] Shaanan Cohney, David A. Hoffman, Jeremy Sklaroff, and David Wishnick. Coin-operated capitalism. *Penn Law: Legal Scholarship Repository*, 119(3), 2019. [Cited on pages 30, 33, and 34]

[CMM+16] Gal Cohensius, Shie Manor, Reshef Meir, Eli Meirom, and Ariel Orda. Proxy voting for better outcomes. *arXiv preprint arXiv:1611.08308*, 2016. [Cited on page 124]

[Coi] CoinMarketCap. Cryptocurrency market capitalizations. `https://coinmarketcap.com`. [Cited on page 1]

[CP93] David Chaum and Torben P. Pedersen. Wallet databases with observers. In *CRYPTO 1992 Proceedings*, volume 740 of *LNCS*, pages 89–105. Springer, 1993. [Cited on pages 9, 22, 78, and 112]

[CR03] Ran Canetti and Tal Rabin. Universal composition with joint state. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*, pages 265–281. Springer, 2003. [Cited on page 23]

[CRS05] David Chaum, Peter Y. A. Ryan, and Steve A. Schneider. A practical voter-verifiable election scheme. In Sabrina De Capitani di Vimercati, Paul F. Syverson, and Dieter Gollmann, editors, *Computer Security - ESORICS 2005, 10th European Symposium on Research in Computer Security, Milan, Italy, September 12-14, 2005, Proceedings*, volume 3679 of *Lecture Notes in Computer Science*, pages 118–139. Springer, 2005. [Cited on pages 59, 60, and 61]

[CS11] Veronique Cortier and Ben Smyth. Attacking and fixing helios: An analysis of ballot secrecy. In *2011 IEEE 24th Computer Security Foundations Symposium*, pages 297–311, 2011. [Cited on page 60]

[CXS+19] Wubing Chen, Zhiying Xu, Shuyu Shi, Yang Zhao, and Jun Zhao. A survey of blockchain applications in different domains. *CoRR*, abs/1911.02013, 2019. [Cited on page 1]

[CZZ+16] Nikos Chondros, Bingsheng Zhang, Thomas Zacharias, Panos Diamantopoulos, Stathis Maneas, Christos Patsonakis, Alex Delis, Aggelos Kiayias, and Mema Roussopoulos. D-demos: A distributed, end-to-end verifiable, internet voting system. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*, pages 711–720, 2016. [Cited on page 61]

[CZZ+19] Nikos Chondros, Bingsheng Zhang, Thomas Zacharias, Panos Diamantopoulos, Stathis Maneas, Christos Patsonakis, Alex Delis, Aggelos Kiayias, and Mema Roussopoulos. Distributed, end-to-end verifiable, and privacy-preserving internet voting systems. *Comput. Secur.*, 83:268–299, 2019. [Cited on page 147]

[Dan] George Danezis. Petlib - a python library that implements a number of privacy enhancing technologies. `https://github.com/gdanezis/petlib`. [Cited on page 151]

[DASa] DASH. Dash is digital cash. https://www.dash.org/. [Cited on pages 1, 3, and 35]

[Dasb] Dash Core Group. Governance. `https://docs.dash.org/en/stable/governance/index.html`. [Cited on pages 3, 27, 35, and 71]

[DD14] Evan Duffield and Daniel Diaz. Dash: A privacy-centric crypto-currency. `https://www.dash.org/wp-content/uploads/2015/04/Dash-WhitepaperV1.pdf`, 2014. [Cited on pages 24, 25, 28, 32, and 71]

[Deca] Decred. Community-directed, superior store of value. `https://decred.org/`. [Cited on pages 3 and 41]

[Decb] Decred. Introduction to decred governance. `https://docs.decred.org/governance/overview/#on-chain-voting`. [Cited on pages 3, 6, 41, 68, 70, 71, and 102]

[Decc] Decred. Voting service providers. `https://decred.org/vsp/`. [Cited on pages 42 and 94]

[Deg] Jonas Degrave. Getopinionated. `https://github.com/getopinionated/getopinionated`. GitHub repository; date last accessed: 2017-10-21. [Cited on page 125]

[Dem17] Democracy Earth. The social smart contract. an open source white paper. `http://democracy.earth`, September 1, 2017. Online; date last accessed: 2017-10-21. [Cited on pages 71 and 125]

[Des94] Yvo G Desmedt. Threshold cryptography. *European Transactions on Telecommunications*, 5(4):449–458, 1994. [Cited on page 56]

[DFL16] Primavera De Filippi and Benjamin Loveluck. The invisible politics of bitcoin: governance crisis of a decentralised infrastructure. *Internet Policy Review*, 5(3), 2016. [Cited on pages 3, 25, and 123]

[DGS03] Ivan Damgård, Jens Groth, and Gorm Salomonsen. The theory and implementation of an electronic voting system. In Dimitris Gritzalis, editor, *Secure Electronic Voting*, volume 7 of *Advances in Information Security*, pages 77–98. Springer, 2003. [Cited on pages 54, 55, 56, 57, 147, and 148]

[DJ01] Ivan Damgård and Mats Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In *Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography: Public Key Cryptography*, PKC '01, pages 119–136, Berlin, Heidelberg, 2001. Springer-Verlag. [Cited on page 57]

[DJN10] Ivan Damgard, Mads Jurik, and Jesper Buus Nielsen. A generalization of paillier's public-key system with applications to electronic voting. *International Journal of Information Security*, 9(6):371–385, 2010. [Cited on page 22]

[DK15] Hans Delfs and Helmut Knebl. *Introduction to Cryptography - Principles and Applications, Third Edition*. Information Security and Cryptography. Springer, 2015. [Cited on page 18]

[DM18] Stephen DiRose and Mo Mansouri. Comparison and analysis of governance mechanisms employed by blockchain-based distributed autonomous organizations. In *2018 13th Annual Conference on System of Systems Engineering (SoSE)*, pages 195–202, 2018. [Cited on page 64]

[dPLC17] Adán Sánchez de Pedro, Daniele Levi, and Luis Iván Cuende. Witnet: A decentralized oracle network protocol. *arXiv preprint arXiv:1711.09756*, 2017. [Cited on page 164]

[DuP17] Quinn DuPont. *Bitcoin and Beyond*, chapter Experiments in algorithmic governance. A history and ethnography of The DAO, a failed decentralized autonomous organization. Routledge, first edition, Nov 2017. [Cited on pages 25 and 123]

[DvdGdSA12] Denise Demirel, Jeroen van de Graaf, and Roberto Samarone dos Santos Araújo. Improving helios with everlasting privacy towards the public. In J. Alex Halderman and Olivier Pereira, editors, *2012 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, EVT/WOTE '12, Bellevue, WA, USA, August 6-7, 2012*. USENIX Association, 2012. [Cited on page 61]

[Elg85] T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985. [Cited on pages 16 and 56]

[ENB11] Kirk Emerson, Tina Nabatchi, and Stephen Balogh. An integrative framework for collaborative governance. *Journal of Public Administration Research and Theory*, 22(1):1–29, May 2011. [Cited on pages 133 and 134]

[Eth20] Etherscan. Token tracker : Erc-20 tokens. `https://etherscan.io/tokens`, Jun 2020. [Cited on page 30]

[FCZ20] Xinxin Fan, Qi Chai, and Zhi Zhong. Multav: A multi-chain token backed voting framework for decentralized blockchain governance. In *International Conference on Blockchain*, pages 33–47. Springer, 2020. [Cited on pages 64 and 65]

[FHF07] Ariel J. Feldman, J. Alex Halderman, and Edward W. Felten. Security analysis of the diebold accuvote-ts voting machine. In Ray Martinez and David A. Wagner, editors, *2007 USENIX/ACCURATE Electronic Voting Technology Workshop, EVT'07, Boston, MA, USA, August 6, 2007*. USENIX Association, 2007. [Cited on page 58]

[FMS10] Jun Furukawa, Kengo Mori, and Kazue Sako. An implementation of a mix-net based network voting scheme and its use in a private organization. In *Towards trustworthy elections*, pages 141–154. Springer, 2010. [Cited on page 55]

[FOO92] Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A practical secret voting scheme for large scale elections. In Jennifer Seberry and Yuliang Zheng, editors, *Advances in Cryptology - AUSCRYPT '92, Workshop on the Theory and Application of Cryptographic Techniques, Gold Coast, Queensland, Australia, December 13-16, 1992, Proceedings*, volume 718 of *Lecture Notes in Computer Science*, pages 244–251. Springer, 1992. [Cited on page 55]

[For02] Bryan Ford. Delegative democracy. *Manuscript*, 2002. [Cited on pages 38, 68, 71, and 124]

[For20] Bryan Ford. A liquid perspective on democratic choice. *CoRR*, abs/2003.12393, 2020. [Cited on page 68]

[Fou] Zcash Foundation. Zcash proposals submission. `https://github.com/ZcashFoundation/GrantProposals-2017Q4/blob/master/README.md`. [Cited on pages 3, 6, 44, 45, 68, 70, 71, and 102]

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO' 86*, pages 186–194, Berlin, Heidelberg, 1986. Springer Berlin Heidelberg. [Cited on page 22]

[Fur04] Jun Furukawa. Efficient, verifiable shuffle decryption and its requirement of unlinkability. In *International Workshop on Public Key Cryptography*, pages 319–332. Springer, 2004. [Cited on page 55]

[GA15] James Green-Armytage. Direct voting and proxy voting. *Constitutional Political Economy*, 26(2):190–220, 2015. [Cited on pages 68 and 124]

[GBE+18] Adem Efe Gencer, Soumya Basu, Ittay Eyal, Robbert van Renesse, and Emin Gün Sirer. Decentralization in bitcoin and ethereum networks. *CoRR*, abs/1801.03998, 2018. [Cited on page 6]

[Get] Getmonero. Monero forum. `https://forum.getmonero.org/`. [Cited on page 31]

[GGR09] Ryan W. Gardner, Sujata Garera, and Aviel D. Rubin. Coercion resistant end-to-end voting. In Roger Dingledine and Philippe Golle, editors, *Financial Cryptography and Data Security, 13th International Conference, FC 2009, Accra Beach, Barbados, February 23-26, 2009. Revised Selected Papers*, volume 5628 of *Lecture Notes in Computer Science*, pages 344–361. Springer, 2009. [Cited on page 60]

[GHM+17] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, China, October 28-31, 2017*, pages 51–68. ACM, 2017. [Cited on pages 2 and 3]

[GJKR99] Rosario Gennaro, Stanisław Jarecki, Hugo Krawczyk, and Tal Rabin. Secure distributed key generation for discrete-log based cryptosystems.

In *EUROCRYPT '99*, pages 295–310, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg. [Cited on pages 73, 81, 86, 109, and 110]

[GK15] Jens Groth and Markulf Kohlweiss. *One-Out-of-Many Proofs: Or How to Leak a Secret and Spend a Coin*, pages 253–280. Springer Berlin Heidelberg, 2015. [Cited on pages 114 and 115]

[GKCC14] Arthur Gervais, Ghassan O. Karame, Vedran Capkun, and Srdjan Capkun. Is bitcoin a decentralized currency? *IEEE Secur. Priv.*, 12(3):54–60, 2014. [Cited on pages 3, 6, and 69]

[GKL15] Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos. The bitcoin backbone protocol: Analysis and applications. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 281–310. Springer, 2015. [Cited on pages 28 and 111]

[GKZ19] Peter Gaži, Aggelos Kiayias, and Dionysis Zindros. Proof-of-stake sidechains. In *IEEE Symposium on Security & Privacy*, 2019. [Cited on pages 2, 24, and 162]

[GM17] Matthew Green and Ian Miers. Bolt: Anonymous payment channels for decentralized currencies. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 473–489. ACM, 2017. [Cited on page 2]

[GMV20] Giancarlo Giudici, Alistair Milne, and Dmitri Vinogradov. Cryptocurrencies: market analysis and perspectives. *Journal of Industrial and Business Economics*, 47(1):1–18, Mar 2020. [Cited on pages 29, 30, 31, and 32]

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, pages 218–229, New York, NY, USA, 1987. ACM. [Cited on page 20]

[Gol04] Oded Goldreich. *The Foundations of Cryptography - Volume 2: Basic Applications*. Cambridge University Press, 2004. [Cited on page 18]

[GPZ17] Panagiotis Grontas, Aris Pagourtzis, and Alexandros Zacharakis. Coercion resistance in a practical secret voting scheme for large scale elections. In *14th International Symposium on Pervasive Systems, Algorithms and Networks & 11th International Conference on Frontier of Computer Science and Technology & Third International Symposium of Creative Computing, ISPAN-FCST-ISCC 2017, Exeter, United Kingdom, June 21-23, 2017*, pages 514–519. IEEE Computer Society, 2017. [Cited on page 55]

[Gri03] Dimitris Gritzalis, editor. *Secure Electronic Voting*, volume 7 of *Advances in Information Security*. Springer, 2003. [Cited on page 148]

[GRTT17] Pasquale Giungato, Roberto Rana, Angela Tarabella, and Caterina Tricase. Current trends in sustainability of bitcoins and related blockchain technology. *Sustainability*, 9(12):2214, Nov 2017. [Cited on page 2]

[GV87] Oded Goldreich and Ronen Vainish. How to solve any protocol problem - an efficiency improvement. In Carl Pomerance, editor, *Advances in Cryptology - CRYPTO '87, A Conference on the Theory and Applications of Cryptographic Techniques, Santa Barbara, California, USA, August 16-20, 1987, Proceedings*, volume 293 of *Lecture Notes in Computer Science*, pages 73–86. Springer, 1987. [Cited on page 23]

[Hac19] Philipp Hacker. Corporate governance for complex cryptocurrencies? a framework for stability and decision making in blockchain-based organizations. *Regulating Blockchain. Techno-Social and Legal Challenges*, pages 140–166, 2019. [Cited on page 25]

[Har16] David Harrison. Decentralized autonomous organizations. `http://www.allenovery.com/SiteCollectionDocuments/Article%20Decentralized%20Autonomous%20Organizations.pdf`, May 16 2016. Online; date last accessed: 2017-10-21. [Cited on page 25]

[HBHW17] Daira Hopwood, Sean Bowe, Tailor Hornby, and Nathan Wilcox. Zcash protocol specification version 2017.0. `https://github.com/zcash/zips/blob/master/protocol/protocol.pdf`, 2017. [Cited on pages 2 and 28]

[Hel06] David Held. *Models of Democracy*. Stanford University Press, Stanford, CA, 3 edition, 2006. [Cited on page 36]

[Her18] Maurice Herlihy. Atomic cross-chain swaps. In Calvin Newport and Idit Keidar, editors, *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC 2018, Egham, United Kingdom, July 23-27, 2018*, pages 245–254. ACM, 2018. [Cited on page 2]

[HHHH18] Friorik P. Hjalmarsson, Gunnlaugur K. Hreioarsson, Mohammad Hamdaqa, and Gísli Hjálmtýsson. Blockchain-based e-voting system. In *11th IEEE International Conference on Cloud Computing, CLOUD 2018, San Francisco, CA, USA, July 2-7, 2018*, pages 983–986. IEEE Computer Society, 2018. [Cited on page 63]

[HL15] Steve Hardt and Lia C. R. Lopes. Google votes: A liquid democracy experiment on a corporate social network. http://www.tdcommons.org/dpubs_series/79/, 2015. [Cited on page 125]

[Hor18] Horizen. Horizen roadmap. dao treasury protocol-level voting system. `https://www.horizen.global/roadmap`, 2018. [Cited on pages 5, 25, 27, and 169]

[HS00] Martin Hirt and Kazue Sako. Efficient receipt-free voting based on homomorphic encryption. In Bart Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 of *Lecture Notes in Computer Science*, pages 539–556. Springer, 2000. [Cited on page 55]

[HS18] Robby Houben and Alexander Snyers. Cryptocurrencies and blockchain: Legal context and implications for financial crime, money laundering and tax evasion, 2018. [Cited on page 29]

[HT18] Philipp Hacker and Chris Thomale. Crypto-securities regulation: Icos, token sales and cryptocurrencies under eu financial law. *European Company and Financial Law Review*, 15(4):645–696, 2018. [Cited on page 30]

[Huf11] Marc Hufty. Investigating policy processes: the governance analytical framework (gaf). *Research for sustainable development: Foundations, experiences, and perspectives*, pages 403–424, 2011. [Cited on page 64]

[HVHC02] E. Herrera-Viedma, F. Herrera, and F. Chiclana. A consensus model for multiperson decision making with different preference structures. *Trans. Sys. Man Cyber. Part A*, 32(3):394–402, may 2002. [Cited on pages 135, 136, and 141]

[HWC+21] Chenyu Huang, Zeyu Wang, Huangxun Chen, Qiwei Hu, Qian Zhang, Wei Wang, and Xia Guan. Repchain: A reputation-based secure, fast, and high incentive blockchain system via sharding. *IEEE Internet Things J.*, 8(6):4291–4304, 2021. [Cited on page 164]

[IB99] Judith E. Innes and David E. Booher. Consensus building and complex adaptive systems. *Journal of the American Planning Association*, 65(4):412–423, 1999. [Cited on pages 133, 134, 135, and 136]

[ICO] ICOAlert. Ico alert - the only complete list of icos, token sales, and crowdsales. calendar of active and upcoming initial coin offerings. `https://www.icoalert.com/`. [Cited on pages 28 and 30]

[IH18] Herminia Ibarra and Morten T. Hansen. Does collaborative leadership have to be slow?, 2018. [Cited on page 135]

[IOH] IOHK. Scorex 2 - the modular blockchain framework. `https://github.com/ScorexFoundation/Scorex`. [Cited on page 151]

[IOH19] IOHK. Treasury prototype implementation. `https://github.com/input-output-hk/TreasuryCoin`, 2019. [Cited on pages 10 and 78]

[IS04] RenÃ©e A. Irvin and John Stansbury. Citizen participation in decision making: Is it worth the effort? *Public Administration Review*, 64(1):55–65, January 2004. [Cited on page 134]

[Ito04] Joichi Ito. Emergent democracy. *Published Online*, 2004. [Cited on page 124]

[JCJ05] Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-resistant electronic elections. In Vijay Atluri, Sabrina De Capitani di Vimercati, and Roger Dingledine, editors, *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society, WPES 2005, Alexandria, VA, USA, November 7, 2005*, pages 61–70. ACM, 2005. [Cited on pages 55 and 62]

[JJR02] Markus Jakobsson, Ari Juels, and Ronald L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In Dan Boneh, editor, *Proceedings of the 11th USENIX Security Symposium, San Francisco, CA, USA, August 5-9, 2002*, pages 339–353. USENIX, 2002. [Cited on page 55]

[JMP13] Hugo Jonker, Sjouke Mauw, and Jun Pang. Privacy and verifiability in voting systems: Methods, developments and trends. *Comput. Sci. Rev.*, 10:1–30, 2013. [Cited on page 58]

[Jon03] Douglas W. Jones. The evaluation of voting technology. In Dimitris Gritzalis, editor, *Secure Electronic Voting*, volume 7 of *Advances in Information Security*, pages 3–16. Springer, 2003. [Cited on page 54]

[KAK18] Kashif Mehboob Khan, Junaid Arshad, and Muhammad Mubashir Khan. Secure digital voting system based on blockchain technology. *Int. J. Electron. Gov. Res.*, 14(1):53–62, 2018. [Cited on pages 57 and 63]

[Kea17] Jonathan Keane. $35 million in 30 seconds: Token sale for internet browser brave sells out, Jun 2017. [Cited on page 30]

[KG17] Rami Khalil and Arthur Gervais. Revive: Rebalancing off-blockchain payment networks. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 439–453. ACM, 2017. [Never cited]

[KJ20] Samika Kashyap and A Jeyasekar. A competent and accurate blockchain based e-voting system on liquid democracy. In *2020 2nd Conference on Blockchain Research Applications for Innovative Networks and Services (BRAINS)*, pages 202–203, 2020. [Cited on page 63]

[KJG+18] Eleftherios Kokoris-Kogias, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ewa Syta, and Bryan Ford. Omniledger: A secure, scale-out, decentralized ledger via sharding. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, pages 583–598. IEEE Computer Society, 2018. [Cited on pages 2 and 24]

[KKH+15] Christoph Carl Kling, Jerome Kunegis, Heinrich Hartmann, Markus Strohmaier, and Steffen Staab. Voting behaviour and power in online democracy: A study of liquidfeedback in germany's pirate party, 2015. [Cited on page 38]

[KKH+17] Christoph Carl Kling, Jerome Kunegis, Heinrich Hartmann, Markus Strohmaier, and Steffen Staab. Federal elections 2016: Election results for the u.s. president, the u.s. senate and the u.s. house of representatives. `https://www.fec.gov/resources/cms-content/documents/federalelections2016.pdf`, dec 2017. [Cited on page 50]

[KKKZ19] Thomas Kerber, Aggelos Kiayias, Markulf Kohlweiss, and Vassilis Zikas. Ouroboros crypsinous: Privacy-preserving proof-of-stake. In *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pages 157–174. IEEE, 2019. [Cited on pages 2, 3, 5, 24, and 124]

[KKN+] Dmytro Kaidalov, Lyudmila Kovalchuk, Andrii Nastenko, Mariia Rodinko, Oleksiy Shevtsov, and Roman Oliynykov. A proposal for an ethereum classic treasury system. `https://iohk.io/research/papers/#AJSEAT7K`. [Cited on pages 25, 27, 72, and 122]

[KKN+17] Dmytro Kaidalov, Lyudmila Kovalchuk, Andrii Nastenko, Mariia Rodinko, Oleksiy Shevtsov, and Roman Oliynykov. Ethereum classic treasury system proposal, 2017. IOHK RESEARCH REPORT. [Cited on pages 6, 25, 27, 28, and 31]

[KKW06] Aggelos Kiayias, Michael Korman, and David Walluck. An internet voting system supporting user privacy. In *22nd Annual Computer Security Applications Conference (ACSAC 2006), 11-15 December 2006, Miami Beach, Florida, USA*, pages 165–174. IEEE Computer Society, 2006. [Cited on page 147]

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. CRC Press, 2014. [Cited on pages 17 and 18]

[KMNV16] Oksana Kulyk, Karola Marky, Stephan Neumann, and Melanie Volkamer. Introducing proxy voting to helios. In *11th International Conference on Availability, Reliability and Security, ARES 2016, Salzburg, Austria, August 31 - September 2, 2016*, pages 98–106. IEEE Computer Society, 2016. [Cited on pages 57, 61, and 62]

[KMP18] Anson Kahng, Simon Mackenzie, and Ariel D. Procaccia. Liquid democracy: An algorithmic perspective. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1095–1102. AAAI Press, 2018. [Cited on page 124]

[KMS+15] Ahmed E. Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. *IACR Cryptology ePrint Archive*, 2015:675, 2015. [Cited on page 29]

[KMS+16] Ahmed E. Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, pages 839–858. IEEE Computer Society, 2016. [Cited on pages 2 and 24]

[KMTZ13] Jonathan Katz, Ueli Maurer, Björn Tackmann, and Vassilis Zikas. Universally composable synchronous computation. In Amit Sahai, editor, *Theory of Cryptography - 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, volume 7785 of *Lecture Notes in Computer Science*, pages 477–498. Springer, 2013. [Cited on page 23]

[KN+] D. Kaidalov, A. Nastenko, et al. Dash governance system: Analysis and suggestions for improvements. `https://iohk.io/research/papers/#NSJ554WR`. [Cited on pages 25, 28, 64, 72, and 102]

[KNM+17] Oksana Kulyk, Stephan Neumann, Karola Marky, Jurlind Budurushi, and Melanie Volkamer. Coercion-resistant proxy voting. *Comput. Secur.*, 71:88–99, 2017. [Cited on page 62]

[KNS+16] Dmytro Kaidalov, Andrii Nastenko, Oleksiy Shevtsov, Mariia Rodinko, Lyudmila Kovalchuk, and Roman Oliynykov. A review of the dash governance system:analysis and suggestions for improvement, 2016. IOHK RESEARCH REPORT. [Cited on pages 6, 25, 27, 31, 32, 35, 36, 37, 41, 66, 70, and 71]

[KR20] Grammateia Kotsialou and Luke Riley. Incentivising participation in liquid democracy with breadth-first delegation. In Amal El Fallah Seghrouchni, Gita Sukthankar, Bo An, and Neil Yorke-Smith, editors, *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*, pages 638–644. International Foundation for Autonomous Agents and Multiagent Systems, 2020. [Cited on page 38]

[KRDO17] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 357–388. Springer, 2017. [Cited on pages 2, 3, 33, and 79]

[KRMC10] John Kelsey, Andrew Regenscheid, Tal Moran, and David Chaum. Attacking paper-based E2E voting systems. In David Chaum, Markus Jakobsson, Ronald L. Rivest, Peter Y. A. Ryan, Josh Benaloh, Miroslaw Kutylowski, and Ben Adida, editors, *Towards Trustworthy Elections, New Directions in Electronic Voting*, volume 6000 of *Lecture Notes in Computer Science*, pages 370–387. Springer, 2010. [Cited on page 60]

[KRS10] Steve Kremer, Mark Ryan, and Ben Smyth. Election verifiability in electronic voting protocols. In Dimitris Gritzalis, Bart Preneel, and Marianthi Theoharidou, editors, *Computer Security - ESORICS 2010, 15th European Symposium on Research in Computer Security, Athens, Greece, September 20-22, 2010. Proceedings*, volume 6345 of *Lecture Notes in Computer Science*, pages 389–404. Springer, 2010. [Cited on page 150]

[KSA+21] Michal Król, Alberto Sonnino, Mustafa Al-Bassam, Argyrios G. Tasiopoulos, Etienne Rivière, and Ioannis Psaras. Proof-of-prestige: A useful work reward system for unverifiable tasks. *ACM Trans. Internet Techn.*, 21(2):44:1–44:27, 2021. [Cited on page 164]

[KSG03] Sepandar D. Kamvar, Mario T. Schlosser, and Hector Garcia-Molina. The eigentrust algorithm for reputation management in P2P networks. In Gusztáv Hencsey, Bebo White, Yih-Farn Robin Chen, László Kovács, and Steve Lawrence, editors, *Proceedings of the Twelfth International World Wide Web Conference, WWW 2003, Budapest, Hungary, May 20-24, 2003*, pages 640–651. ACM, 2003. [Cited on page 164]

[KSRW04] Tadayoshi Kohno, Adam Stubblefield, Aviel D. Rubin, and Dan S. Wallach. Analysis of an electronic voting system. In *2004 IEEE Symposium on Security and Privacy (S&P 2004), 9-12 May 2004, Berkeley, CA, USA*, page 27. IEEE Computer Society, 2004. [Cited on page 58]

[KSW05] Chris Karlof, Naveen Sastry, and David A. Wagner. Cryptographic voting protocols: A systems perspective. In Patrick D. McDaniel, editor, *Proceedings of the 14th USENIX Security Symposium, Baltimore, MD, USA, July 31 - August 5, 2005*. USENIX Association, 2005. [Cited on page 60]

[KTV15] Oksana Kulyk, Vanessa Teague, and Melanie Volkamer. Extending helios towards private eligibility verifiability. In Rolf Haenni, Reto E. Koenig, and Douglas Wikström, editors, *E-Voting and Identity - 5th International Conference, VoteID 2015, Bern, Switzerland, September 2-4, 2015, Proceedings*, volume 9269 of *Lecture Notes in Computer Science*, pages 57–73. Springer, 2015. [Cited on page 61]

[KY04] Aggelos Kiayias and Moti Yung. The vector-ballot e-voting approach. In Ari Juels, editor, *Financial Cryptography, 8th International Conference, FC 2004, Key West, FL, USA, February 9-12, 2004. Revised Papers*, volume 3110 of *Lecture Notes in Computer Science*, pages 72–89. Springer, 2004. [Cited on pages 147 and 148]

[KZ19] Aggelos Kiayias and Dionysis Zindros. Proof-of-work sidechains. In *Financial Cryptography and Data Security - FC 2019 International Workshops, VOTING and WTSC, St. Kitts, St. Kitts and Nevis, February 18-22, 2019, Revised Selected Papers*, volume 11599 of *Lecture Notes in Computer Science*, pages 21–34. Springer, 2019. [Cited on pages 2, 24, and 162]

[KZZ15a] Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. DEMOS-2: scalable E2E verifiable elections without random oracles. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, pages 352–363. ACM, 2015. [Cited on pages 59 and 61]

[KZZ15b] Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. End-to-end verifiable elections in the standard model. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 468–498. Springer, 2015. [Cited on pages 58, 60, and 61]

[Lew94] James R. Lewis. Sample sizes for usability studies: Additional considerations. *Human Factors*, 36(2):368–378, 1994. PMID: 8070799. [Cited on page 137]

[LGT+03] Costas Lambrinoudakis, Dimitris Gritzalis, Vassilis Tsoumas, Maria Karyda, and Spyros Ikonomopoulos. Secure electronic voting: the current landscape. In Dimitris Gritzalis, editor, *Secure Electronic Voting*, volume 7 of *Advances in Information Security*, pages 101–122. Springer, 2003. [Cited on pages 56 and 57]

[LIJ84] AREND LIJPHART. *Democracies: Patterns of Majoritarian and Consensus Government in Twenty-One Countries*. Yale University Press, 1984. [Cited on page 36]

[Lin09] Yehuda Lindell. General composition and universal composability in secure multiparty computation. *J. Cryptology*, 22(3):395–428, 2009. [Cited on page 23]

[Liq] LiquidFeedback. LiquidFeedback official website. `http://liquidfeedback.org`. Online; date last accessed: 2017-10-21. [Cited on page 125]

[LJEK16] Kibin Lee, Joshua James, Tekachew Ejeta, and Hyoung Kim. Electronic voting service using block-chain. *Journal of Digital Forensics, Security and Law*, 2016. [Cited on pages 71 and 125]

[LLZ+21] Yue Liu, Qinghua Lu, Liming Zhu, Hye-Young Paik, and Mark Staples. A systematic literature review on blockchain governance. *CoRR*, abs/2105.05460, 2021. [Cited on pages 64 and 65]

[Lom03] Dennis Lomax. Beyond politics, an introduction, January 1 2003. Online; date last accessed: 2017-10-21. [Cited on page 124]

[LXS+19] Zhuotao Liu, Yangxi Xiang, Jian Shi, Peng Gao, Haoyu Wang, Xusheng Xiao, Bihan Wen, and Yih-Chun Hu. Hyperservice: Interoperability and programmability across heterogeneous blockchains. In *Proceedings of*

the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019, pages 549–566. ACM, 2019. [Cited on pages 1, 2, and 24]

[MA18] Mahdi H. Miraz and Maaruf Ali. Applications of blockchain technology beyond cryptocurrency. *CoRR*, abs/1801.03528, 2018. [Cited on page 6]

[MARP20] Paul Merrill, Thomas H Austin, Justin Rietz, and Jon Pearce. Ping-pong governance: Token locking for enabling blockchain self-governance. In *Mathematical Research for Blockchain Economy*, pages 13–29. Springer, 2020. [Cited on page 64]

[Max13] Gregory Maxwell. Coinjoin: Bitcoin privacy for the real world. `https://bitcointalk.org/index.php?topic=279249.0`, 2013. [Cited on pages 24 and 123]

[MBB+19] Andrew Miller, Iddo Bentov, Surya Bakshi, Ranjit Kumaresan, and Patrick McCorry. Sprites and state channels: Payment networks that go faster than lightning. In *Financial Cryptography and Data Security - 23rd International Conference, FC 2019, Frigate Bay, St. Kitts and Nevis, February 18-22, 2019, Revised Selected Papers*, pages 508–526. Springer, 2019. [Cited on page 2]

[MC13] Tyler Moore and Nicolas Christin. Beware the middleman: Empirical analysis of bitcoin-exchange risk. In *Financial Cryptography and Data Security - 17th International Conference, FC 2013, Okinawa, Japan, April 1-5, 2013, Revised Selected Papers*, pages 25–33. Springer, 2013. [Cited on page 3]

[McC16] Smàri McCarthy. Wasa2il. `https://github.com/smari/wasa2il`, 2016. GitHub repository; date last accessed: 2017-10-21. [Cited on pages 71 and 125]

[MCLH19] Thomas McGhin, Kim-Kwang Raymond Choo, Charles Zhechao Liu, and Debiao He. Blockchain in healthcare applications: Research challenges and opportunities. *Journal of Network and Computer Applications*, 135:62 – 75, 2019. [Cited on page 1]

[Mer16] R Merkle. Daos, democracy and governance. `http://merkle.com/papers/DAOdemocracyDraft.pdf`, 2016. [Cited on page 25]

[MGK02] Lilian Mitrou, Dimitris Gritzalis, and Sokratis K. Katsikas. Revisiting legal and regulatory requirements for secure e-voting. In Adeeb Ghonaimy, Mahmoud T. El-Hadidi, and Heba Kamal Aslan, editors, *Security in the Information Society: Visions and Perspectives, IFIP TC11 17th International Conference on Information Security (SEC2002), May 7-9, 2002, Cairo, Egypt*, volume 214 of *IFIP Conference Proceedings*, pages 469–480. Kluwer, 2002. [Cited on page 57]

[MGKQ03] Lilian Mitrou, Dimitris Gritzalis, Sokratis K. Katsikas, and Gerald Quirchmayr. Electronic voting: Constitutional and legal requirements, and their

technical implications. In Dimitris Gritzalis, editor, *Secure Electronic Voting*, volume 7 of *Advances in Information Security*, pages 43–60. Springer, 2003. [Cited on page 57]

[MMAM+16] Trent McConaghy, Rodolphe Marques, Dimitri Andreas Muller, De Jonghe, Troy McConaghy, Greg McMullen, Ryan Henderson, Sylvain Bellemare, and Alberto Granzotto. Bigchaindb: a scalable blockchain database. *whitepaper*, 2016. [Cited on page 2]

[MMK+17] Giulio Malavolta, Pedro Moreno-Sanchez, Aniket Kate, Matteo Maffei, and Srivatsan Ravi. Concurrency and privacy with payment-channel networks. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 455–471. ACM, 2017. [Cited on page 2]

[MMZ+20] Deepak Maram, Harjasleen Malvai, Fan Zhang, Nerla Jean-Louis, Alexander Frolov, Tyler Kell, Tyrone Lobban, Christine Moy, Ari Juels, and Andrew Miller. Candid: Can-do decentralized identity with legacy compatibility, sybil-resistance, and accountability. *IACR Cryptol. ePrint Arch.*, 2020:934, 2020. [Cited on page 169]

[Mol01] Richard A. Mollin. *An introduction to cryptography*. CRC Press series on discrete mathematics and its applications. Chapman&Hall/CRC Press, 2001. [Cited on page 18]

[Mon] Monero. The monero project. `https://www.getmonero.org`. [Cited on pages 2 and 31]

[MPG+20] Lawrence Mosley, Hieu Pham, Xiaoshi Guo, Yogesh Bansal, Eric Hare, and Nadia Antony. Towards a systematic understanding of blockchain governance in proposal voting: A dash case study. *Available at SSRN 3416564*, 2020. [Cited on page 64]

[MSA19] Ahmed Afif Monrat, Olov Schelén, and Karl Andersson. A survey of blockchain from the perspectives of applications, challenges, and opportunities. *IEEE Access*, 7:117134–117151, 2019. [Cited on pages 26 and 30]

[MSH+16] Ujan Mukhopadhyay, Anthony Skjellum, Oluwakemi Hambolu, Jon Oakley, Lu Yu, and Richard R. Brooks. A brief survey of cryptocurrency systems. In *14th Annual Conference on Privacy, Security and Trust, PST 2016, Auckland, New Zealand, December 12-14, 2016*, pages 745–752. IEEE, 2016. [Cited on page 1]

[MSH17] Patrick McCorry, Siamak F. Shahandashti, and Feng Hao. A smart contract for boardroom voting with maximum voter privacy. In Aggelos Kiayias, editor, *Financial Cryptography and Data Security - 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers*, volume 10322 of *Lecture Notes in Computer Science*, pages 357–375. Springer, 2017. [Cited on pages 61, 63, and 144]

[MZZS18] Gianluca Miscione, Rafael Ziolkowski, Liudmila Zavolokina, and Gerhard Schwabe. Tribal governance: The business of blockchain authentication. In Tung Bui, editor, *51st Hawaii International Conference on System Sciences, HICSS 2018, Hilton Waikoloa Village, Hawaii, USA, January 3-6, 2018*, pages 1–10. ScholarSpace / AIS Electronic Library (AISeL), 2018. [Cited on page 122]

[NAH16] Till Neudecker, Philipp Andelfinger, and Hannes Hartenstein. Timing analysis for inferring the topology of the bitcoin peer-to-peer network. In *Proceedings of the 13th IEEE International Conference on Advanced and Trusted Computing (ATC)*, July 2016. [Cited on page 164]

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. `https://bitcoin.org/bitcoin.pdf`, 2008. [Cited on pages 1, 24, 26, 28, 33, and 79]

[NBF+16] Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder. *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton University Press, Princeton, NJ, USA, 2016. [Cited on pages 1, 28, 33, 79, and 123]

[Nef01] C. Andrew Neff. A verifiable secret shuffle and its application to e-voting. In Michael K. Reiter and Pierangela Samarati, editors, *CCS 2001, Proceedings of the 8th ACM Conference on Computer and Communications Security, Philadelphia, Pennsylvania, USA, November 6-8, 2001*, pages 116–125. ACM, 2001. [Cited on page 55]

[Nef04] C. Andrew Neff. Practical high certainty intent verification for encrypted votes, 2004. [Cited on pages 58, 60, and 61]

[Nor03] Mikael Nordfors. Democracy 2.1: How to make a bunch of lazy and selfish people work together, 2003. Online; date last accessed: 2017-10-21. [Cited on page 124]

[nVo17] nVotes. 3 crypto schemes for liquid democracy (iii). `https://nvotes.com/3-crypto-schemes-liquid-democracy-iii/`, July 19 2017. Online; date last accessed: 2017-10-21. [Cited on pages 71 and 125]

[OA00] Miyako Ohkubo and Masayuki Abe. A length-invariant hybrid mix. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 178–191. Springer, 2000. [Cited on page 55]

[OEC19] OECD. Initial coin offerings (icos) for sme financing. `http://www.oecd.org/finance/ICOs-for-SME-Financing.pdf`, Jan 2019. [Cited on page 30]

[Oka97] Tatsuaki Okamoto. Receipt-free electronic voting schemes for large scale elections. In Bruce Christianson, Bruno Crispo, T. Mark A. Lomas, and Michael Roe, editors, *Security Protocols, 5th International Workshop, Paris, France, April 7-9, 1997, Proceedings*, volume 1361 of *Lecture Notes in Computer Science*, pages 25–35. Springer, 1997. [Cited on page 55]

[OMA+99] Miyako Ohkubo, Fumiaki Miura, Masayuki Abe, Atsushi Fujioka, and Tatsuaki Okamoto. An improvement on a practical secret voting scheme. In *International Workshop on Information Security*, pages 225–234. Springer, 1999. [Cited on page 55]

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 1999. [Cited on page 56]

[PD16] Joseph Poon and Thaddeus Dryja. The bitcoin lightning network: Scalable off-chain instant payments, 2016. [Cited on page 2]

[Ped91] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991. [Cited on page 19]

[PH10] Stefan Popoveniuc and Benjamin Hosp. An introduction to punchscan. In David Chaum, Markus Jakobsson, Ronald L. Rivest, Peter Y. A. Ryan, Josh Benaloh, Miroslaw Kutylowski, and Ben Adida, editors, *Towards Trustworthy Elections, New Directions in Electronic Voting*, volume 6000 of *Lecture Notes in Computer Science*, pages 242–259. Springer, 2010. [Cited on pages 59, 60, and 61]

[PHM95] H Petersen, P Horster, and M Michels. Blind multisignature schemes and their relevance to electronic voting. In *11th Annual Computer Security Applications Conference, IEEE Press*, pages 149–155. Citeseer, 1995. [Cited on page 55]

[PIK93] Choonsik Park, Kazutomo Itoh, and Kaoru Kurosawa. Efficient anonymous channel and all/nothing election scheme. In Tor Helleseth, editor, *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, volume 765 of *Lecture Notes in Computer Science*, pages 248–259. Springer, 1993. [Cited on pages 55 and 57]

[PJ16] Colin Percival and Simon Josefsson. The scrypt password-based key derivation function. *RFC*, 7914:1–16, 2016. [Cited on page 2]

[PK15] Jack Peterson and Joseph Krug. Augur: a decentralized, open-source platform for prediction markets. *arXiv preprint arXiv:1501.01042*, 2015. [Cited on page 164]

[PKRV10] Stefan Popoveniuc, John Kelsey, Andrew Regenscheid, and Poorvi L. Vora. Performance requirements for end-to-end verifiable elections. In

Douglas W. Jones, Jean-Jacques Quisquater, and Eric Rescorla, editors, *2010 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, EVT/WOTE '10, Washington, D.C., USA, August 9-10, 2010*. USENIX Association, 2010. [Cited on pages 58, 59, and 60]

[PP10] Christof Paar and Jan Pelzl. *Understanding Cryptography - A Textbook for Students and Practitioners*. Springer, 2010. [Cited on page 18]

[Pro15] Proxy.me. Voteflow. `http://www.proxyfor.me/help`, 2015. Online; date last accessed: 2017-10-21. [Cited on page 125]

[PSF19] Serguei Popov, Olivia Saa, and Paulo Finardi. Equilibria in the tangle. *Computers & Industrial Engineering*, 136:160–172, Oct 2019. [Cited on page 2]

[PSSK20] Mehrdokht Pournader, Yangyan Shi, Stefan Seuring, and SC Lenny Koh. Blockchain applications in supply chains, transport and logistics: a systematic review of the literature. *International Journal of Production Research*, 58(7):2063–2081, 2020. [Cited on page 6]

[Rei17] Christian Reitwiessner. Ethereum improvement proposal 196 - precompiled contracts for addition and scalar multiplication on the elliptic curve alt bn128. `https://github.com/ethereum/EIPs/blob/master/EIPS/eip-196.md`, 2017. [Cited on page 145]

[Rya08] Peter Y. A. Ryan. Prêt à voter with paillier encryption. *Math. Comput. Model.*, 48(9-10):1646–1662, 2008. [Cited on page 63]

[SB96] R. Schmitt-Beck. Mass media, the electorate, and the bandwagon. a study of communication effects on vote choice in germany. *International Journal of Public Opinion Research*, 8(3):266–291, 1996. [Cited on page 43]

[Sch80] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, oct 1980. [Cited on page 20]

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3):161–174, 1991. [Cited on page 22]

[Sch99] Berry Schoenmakers. A simple publicly verifiable secret sharing scheme and its application to electronic. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 148–164. Springer, 1999. [Cited on page 56]

[SEA19] Microsoft SEAL (release 3.4), oct 2019. Microsoft Research, Redmond, WA. [Cited on page 170]

[SeMS+20] Koen Smit, Jalal el Mansouri, Sabri Saïd, John van Meerten, and Sam Leewis. Decision rights and governance within the blockchain domain: a literature analysis. In *Pacific Asia Conference on Information Systems, PACIS*, 2020. [Cited on page 65]

[SFD+14] Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine, and J. Alex Halderman. Security analysis of the estonian internet voting system. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, pages 703–715. ACM, 2014. [Cited on page 58]

[SGY20] Mohamed Seifelnasr, Hisham S. Galal, and Amr M. Youssef. Scalable open-vote network on ethereum. In Matthew Bernhard, Andrea Bracciali, L. Jean Camp, Shin'ichiro Matsuo, Alana Maurushat, Peter B. Rønne, and Massimiliano Sala, editors, *Financial Cryptography and Data Security - FC 2020 International Workshops, AsiaUSEC, CoDeFi, VOTING, and WTSC, Kota Kinabalu, Malaysia, February 14, 2020, Revised Selected Papers*, volume 12063 of *Lecture Notes in Computer Science*, pages 436–450. Springer, 2020. [Cited on pages 63 and 145]

[SK95] Kazue Sako and Joe Kilian. Receipt-free mix-type voting scheme - A practical solution to the implementation of a voting booth. In Louis C. Guillou and Jean-Jacques Quisquater, editors, *Advances in Cryptology - EUROCRYPT '95, International Conference on the Theory and Application of Cryptographic Techniques, Saint-Malo, France, May 21-25, 1995, Proceeding*, volume 921 of *Lecture Notes in Computer Science*, pages 393–403. Springer, 1995. [Cited on pages 55 and 57]

[Sma03] N.P. Smart. *Cryptography: An Introduction*. Mcgraw-hill education. McGraw-Hill, 2003. [Cited on page 18]

[SP06] Krishna Sampigethaya and Radha Poovendran. A framework and taxonomy for comparison of electronic voting schemes. *Comput. Secur.*, 25(2):137–153, 2006. [Cited on pages 55 and 58]

[SV14] Nigel P. Smart and Frederik Vercauteren. Fully homomorphic SIMD operations. *Des. Codes Cryptogr.*, 71(1):57–81, 2014. [Cited on page 170]

[Szt15] Paul Sztorc. Truthcoin, peer-to-peer oracle system and prediction marketplace, 2015. [Cited on page 164]

[Tak18] Ikuya Takashima. Litecoin: The ultimate guide to the world of litecoin, 2018. [Cited on pages 2 and 24]

[Tez] Tezos. Secure. upgradable. built to last. `https://tezos.com`. [Cited on page 28]

[Tor05] Benno Torgler. Tax morale and direct democracy. *European Journal of Political Economy*, 21(2):525 – 531, 2005. [Cited on page 36]

[Tri89] John Trimbur. Consensus and difference in collaborative learning. *College English*, 51(6):602, 1989. [Cited on page 134]

[Vat00] Adrian Vatter. Consensus and direct democracy: Conceptual and empirical linkages. *European Journal of Political Research*, 38(2):171–192, 2000. [Cited on page 36]

[VB15] Fabian Vogelsteller and Vitalik Buterin. Eip-20: Erc-20 token standard. `https://eips.ethereum.org/EIPS/eip-20`, November 2015. [Cited on page 30]

[Vir92] Robert A. Virzi. Refining the test phase of usability evaluation: How many subjects is enough? *Hum. Factors*, 34(4):457–468, Aug. 1992 1992. [Cited on page 137]

[Vra17] Harald Vranken. Sustainability of bitcoin and blockchains. *Current Opinion in Environmental Sustainability*, 28:1 – 9, 2017. Sustainability governance. [Cited on page 2]

[vS13] Nicolas van Saberhagen. Cryptonote v 2.0. `https://cryptonote.org/whitepaper.pdf`, 2013. [Cited on pages 2, 5, and 123]

[VVL17] Robert Viglione, Rolf Versluis, and Jane Lippencott. Zen white paper. `https://zensystem.io/assets/Zen%20White%20Paper.pdf`, 2017. [Cited on pages 1, 38, 39, and 40]

[WA17] Carl Watner and Bonus Army. Introducing the 'do nothing technologies' blockchain-based ico: Featured bitcoin news, Jun 2017. [Cited on page 30]

[WHO19] BITCOIN WHOS WHO. Bitcoin address lookup, checker and alerts, 2019. [Cited on page 123]

[Wik04] Douglas Wikström. Universally composable DKG with linear number of exponentiations. In *SCN 2004*, pages 263–277. Springer Berlin Heidelberg, 2004. [Cited on pages 85 and 106]

[Woo14] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. `https://www.ethereum.github.io/yellowpaper/paper.pdf`, 2014. [Cited on pages 2, 24, 25, and 48]

[WS06] Kevin Walsh and Emin Gün Sirer. Experience with an object reputation system for peer-to-peer filesharing. In Larry L. Peterson and Timothy Roscoe, editors, *3rd Symposium on Networked Systems Design and Implementation (NSDI 2006), May 8-10, 2007, San Jose, California, USA, Proceedings*. USENIX, 2006. [Cited on page 164]

[WSNH19] Gang Wang, Zhijie Jerry Shi, Mark Nixon, and Song Han. Sok: Sharding on blockchain. In *Proceedings of the 1st ACM Conference on Advances in Financial Technologies, AFT 2019, Zurich, Switzerland, October 21-23, 2019*, pages 41–61. ACM, 2019. [Cited on page 2]

[WWH+10] Scott Wolchok, Eric Wustrow, J. Alex Halderman, Hari K. Prasad, Arun Kankipati, Sai Krishna Sakhamuri, Vasavya Yagati, and Rop Gonggrijp. Security analysis of india's electronic voting machines. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010*, pages 1–14. ACM, 2010. [Cited on page 58]

[WZN+19] Xu Wang, Xuan Zha, Wei Ni, Ren Ping Liu, Y. Jay Guo, Xinxin Niu, and Kangfeng Zheng. Survey on blockchain for internet of things. *Comput. Commun.*, 136:10–29, 2019. [Cited on page 1]

[WZX+18] Huaimin Wang, Zibin Zheng, Shaoan Xie, Hong-Ning Dai, and Xiangping Chen. Blockchain challenges and opportunities: a survey. *International Journal of Web and Grid Services*, 14:352 – 375, 2018. [Cited on page 26]

[XSHT08] Zhe Xia, Steve A. Schneider, James Heather, and Jacques Traoré. Analysis, improvement, and simplification of prêt à voter with paillier encryption. In David L. Dill and Tadayoshi Kohno, editors, *2008 USENIX/ACCURATE Electronic Voting Workshop, EVT 2008, July 28-29, 2008, San Jose, CA, USA, Proceedings*. USENIX Association, 2008. [Cited on page 59]

[XTH+19] Jun-feng Xie, Helen Tang, Tao Huang, F. Richard Yu, Renchao Xie, Jiang Liu, and Yunjie Liu. A survey of blockchain technology applied to smart cities: Research issues and challenges. *IEEE Commun. Surv. Tutorials*, 21(3):2794–2830, 2019. [Cited on page 1]

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 160–164. IEEE Computer Society, 1982. [Cited on page 54]

[YF94] RONALD R. YAGER and DIMITAR P. FILEV. Parameterized and-uke and or-like owa operators. *International Journal of General Systems*, 22(3):297–316, 1994. [Cited on page 141]

[YWY+20] Guangsheng Yu, Xu Wang, Kan Yu, Wei Ni, J. Andrew Zhang, and Ren Ping Liu. Survey: Sharding in blockchains. *IEEE Access*, 8:14155–14181, 2020. [Cited on page 2]

[ZB18] Bingsheng Zhang and Hamed Balogun. On the sustainability of blockchain funding. In Hanghang Tong, Zhenhui Jessie Li, Feida Zhu, and Jeffrey Yu, editors, *2018 IEEE International Conference on Data Mining Workshops, ICDM Workshops, Singapore, Singapore, November 17-20, 2018*, pages 89–96. IEEE, 2018. [Cited on pages 27 and 70]

[Zca] Zcash. Zcash - all coins are created equal. [Cited on pages 1, 2, 3, 32, and 44]

[ZCC+13] Filip Zagórski, Richard Carback, David Chaum, Jeremy Clark, Aleksander Essex, and Poorvi L. Vora. Remotegrity: Design and use of an end-to-end verifiable remote voting system. In Michael J. Jacobson Jr., Michael E. Locasto, Payman Mohassel, and Reihaneh Safavi-Naini, editors, *Applied Cryptography and Network Security - 11th International Conference, ACNS 2013, Banff, AB, Canada, June 25-28, 2013. Proceedings*, volume 7954 of *Lecture Notes in Computer Science*, pages 441–457. Springer, 2013. [Cited on pages 60 and 61]

[ZCC+16a] Fan Zhang, Ethan Cecchetti, Kyle Croman, Ari Juels, and Elaine Shi. Town crier: An authenticated data feed for smart contracts. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 270–282. ACM, 2016. [Cited on page 2]

[ZCC+16b] Fan Zhang, Ethan Cecchetti, Kyle Croman, Ari Juels, and Elaine Shi. Town crier: An authenticated data feed for smart contracts. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 270–282. ACM, 2016. [Cited on page 143]

[ZHS19] Markos Zachariadis, Garrick Hileman, and Susan V Scott. Governance and control in distributed ledgers: Understanding the challenges facing blockchain technology in financial services. *Information and Organization*, 29(2):105–117, 2019. [Cited on page 65]

[ZHT13] Bernd Zwattendorfer, Christoph Hillebold, and Peter Teufl. Secure and privacy-preserving proxy voting system. In *IEEE 10th International Conference on e-Business Engineering, ICEBE 2013, Coventry, United Kingdom, September 11-13, 2013*, pages 472–477. IEEE Computer Society, 2013. [Cited on page 62]

[ZMR18] Mahdi Zamani, Mahnush Movahedi, and Mariana Raykova. Rapidchain: Scaling blockchain via full sharding. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 931–948. ACM, 2018. [Cited on pages 2 and 24]

[ZMS20] Rafael Ziolkowski, Gianluca Miscione, and Gerhard Schwabe. Decision problems in blockchain governance: Old wine in new bottles or walking in someone else's shoes? *Journal of Management Information Systems*, 37(2):316–348, 2020. [Cited on page 65]

[ZOB18] Bingsheng Zhang, Roman Oliynykov, and Hamed Balogun. A treasury system for cryptocurrencies: Enabling better collaborative intelligence. *IACR Cryptology ePrint Archive*, 2018:435, 2018. [Cited on pages 12, 25, 27, 28, and 122]

[ZOB19] Bingsheng Zhang, Roman Oliynykov, and Hamed Balogun. A treasury system for cryptocurrencies: Enabling better collaborative intelligence. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society, 2019. [Cited on pages 28, 31, 33, and 70]

[ZPMS19] Rafael Ziolkowski, Geetha Parangi, Gianluca Miscione, and Gerhard Schwabe. Examining gentle rivalry: Decision-making in blockchain systems.

In Tung Bui, editor, *52nd Hawaii International Conference on System Sciences, HICSS 2019, Grand Wailea, Maui, Hawaii, USA, January 8-11, 2019*, pages 1–10. ScholarSpace, 2019. [Cited on page 122]

[ZS18] Kaspars Zile and Renate Strazdina. Blockchain use cases and their feasibility. *Appl. Comput. Syst.*, 23(1):12–20, 2018. [Cited on page 1]

[ZZ17a] Bingsheng Zhang and Hong-Sheng Zhou. Brief announcement: Statement voting and liquid democracy. In *Proceedings of the ACM Symposium on Principles of Distributed Computing, PODC 2017, Washington, DC, USA, July 25-27, 2017*, pages 359–361, 2017. [Cited on pages 38, 124, and 125]

[ZZ17b] Bingsheng Zhang and Hong-Sheng Zhou. Statement voting. `https://eprint.iacr.org/2017/616`, 2017. [Cited on pages 124 and 125]

Appendices

In Chapter 3 and Chapter 4, we provide the details of our blockchain-based treasury system and privacy-preserving general decision-making system. Recall that both systems rely on the delegable stake-weighted voting scheme for their operations. Therefore, this appendix presents the relevant zero-knowledge proof protocols used to guarantee that participants (voters, experts and voting committee members) in the systems behave honestly in the protocol execution. Finally, Appendix B provides the security proof for the voting protocol at the core of our systems proposed in this thesis.

A Non-interactive Zero-knowledge Proofs Protocols

A.1 NIZK for Lifted Elgamal Encryption of 0

Non-Interactive Zero Knowledge proof for Lifted-Elgamal Encryption of 0

$$\mathsf{NIZK}\{(\mathsf{pk}, C), (m, r) : C = \mathsf{EC.Enc}_{\mathsf{pk}}(m; r) \wedge m = 0\}$$

Statement: Public key, $\mathsf{pk} := h \in \mathbb{G}$, and ciphertext $C := (C_1, C_2) = (g^r, g^m h^r)$
Witness: $m = 0$ and $r \in \mathbb{Z}_p$

Prover:

- Pick random $w \leftarrow \mathbb{Z}_p$; Compute $A_1 := g^w$ and $A_2 := h^w$
- Compute $e_1 = \mathsf{hash}(h, C, A_1, A_2)$ and $z = r * e_1 + w$
- Return $\pi_1 := (A_1, A_2, z_1)$

Verifier:

- Compute $e_1 = \mathsf{hash}(h, C, A_1, A_2)$ and return valid if and only if
  - $g^{z_1} = {C_1}^{e_1} \cdot A_1$
  - $h^{z_1} = {C_2}^{e_1} \cdot A_2$

Figure 1: Non-Interactive Zero Knowledge proof for Lifted-Elgamal Encryption of 0

A.2 NIZK for Lifted Elgamal Encryption of 1

Non-Interactive Zero Knowledge proof for Lifted-Elgamal encryption of 1

$$\mathsf{NIZK}\{(\mathsf{pk}, C), (m, r) : C = \mathsf{EC.Enc}_{\mathsf{pk}}(m; r) \wedge m = 1\}$$

Statement: Public key, $\mathsf{pk} := h \in \mathbb{G}$, and ciphertext $C := (C_1, C_2) = (g^r, g^m h^r)$
Witness: $m = 1$ and $r \in \mathbb{Z}_p$

Prover:

- Pick random $v \leftarrow \mathbb{Z}_p$; Compute $B_1 := g^v$ and $B_2 := h^v$
- Compute $e_2 = \mathsf{hash}(h, C, B_1, B_2)$ and $z_2 = r * e_2 + v$
- Return $\pi_2 := (B_1, B_2, z_2)$

Verifier:

- Compute $e_2 = \mathsf{hash}(h, C, B_1, B_2)$ and return valid if and only if
  - $g^{z_2} = {C_1}^{e_2} \cdot B_1$
  - $h^{z_2} = (C_2 \cdot g^{-1})^{e_2} \cdot B_2$

Figure 2: Non-Interactive Zero Knowledge proof for Lifted-Elgamal encryption of 1

A.3 NIZK for Lifted Elgamal Encryption of 0 or 1

Non-Interactive Zero Knowledge proof for Lifted-Elgamal encryption of 0 or 1

$$\mathsf{NIZK}\{(\mathsf{pk}, C), (m, r) : C = \mathsf{EC.Enc}_{\mathsf{pk}}(m; r) \wedge m \in \{0, 1\}\}$$

Statement: Public key, $\mathsf{pk} := h \in \mathbb{G}$, and ciphertext $C := (C_1, C_2) = (g^r, g^m h^r)$
Witness: $m \in \{0, 1\}$ and $r \in \mathbb{Z}_p$

Prover:

- if m=0:
  - Pick random $w \leftarrow \mathbb{Z}_p$; Compute $A_1 := g^w$ and $A_2 := h^w$
  - Pick random $z_2 \leftarrow \mathbb{Z}_p, e_2 \leftarrow \{0, 1\}^{256}$; Compute $B_1 := \frac{g^{z_2}}{{C_1}^{e_2}}$ and $B_2 := \frac{h^{z_2}}{(\frac{C_2}{g})^{e_2}}$
  - Compute $e := \mathsf{hash}(C, h, A_1, A_2, B_1, B_2)$ and $e_1 := e \oplus e_2$
  - Compute $z_1 := r \cdot e_1 + w$
  - Return: $\pi_3 := (A_1, A_2, B_1, B_2, e_1, e_2, z_1, z_2)$
- else m=1:

- Pick random $v \leftarrow \mathbb{Z}_p$; Compute $B_1 := g^v$ and $B_2 := h^v$
- Pick random $z_1 \leftarrow \mathbb{Z}_p, e_1 \leftarrow \{0,1\}^{256}$; Compute $A_1 := g^{z_1} \cdot {C_1}^{-e_1}$ and $A_2 := \frac{h^{z_1}}{{C_2}^{e_1}}$
- Compute $e := \mathsf{hash}(C, h, A_1, A_2, B_1, B_2)$ and $e_2 := e \oplus e_1$
- Compute $z_2 := r \cdot e_2 + v$
- Return: $\pi_3 := (A_1, A_2, B_1, B_2, e_1, e_2, z_1, z_2)$

Verifier:

- Computer $e = \mathsf{hash}(C, h, A_1, A_2, B_1, B_2)$ and return valid if and only if
  - $e = e_1 \oplus e_2$
  - ${g^{z_1}} = {C_1}^{e_1} \cdot A_1$
  - ${h^{z_1}} = {C_2}^{e_1} \cdot A_2$
  - ${g^{z_2}} = {C_1}^{e_2} \cdot B_1$
  - ${h^{z_2}} = (\frac{C_2}{g})^{e_2} \cdot B_2$

Figure 3: Non-Interactive Zero Knowledge proof for Lifted-Elgamal encryption of 0 or 1

A.4 Alternative NIZK for Lifted Elgamal Encryption of $0/1$

Non-Interactive Zero Knowledge proof for lifted Elgamal Encryption of $0/1$

$$\mathsf{NIZK}\{(\mathsf{pk}, C), (m, r) : C = \mathsf{EC.Enc}_{\mathsf{pk}}(m; r) \wedge m \in \{0, 1\}\}$$

Statement: Public key, $\mathsf{pk} := h \in \mathbb{G}$, and ciphertext $C := (C_1, C_2) = (g^r, g^m h^r)$
Witness: $m \in \{0, 1\}$ and $r \in \mathbb{Z}_p$

Prover:

- Pick random $\beta, \gamma, \delta \leftarrow \mathbb{Z}_p$; Compute $B = \mathsf{EC.Enc}_{\mathsf{pk}}(\beta; \gamma)$ and $A = \mathsf{EC.Enc}_{\mathsf{pk}}(m \cdot \beta; \delta)$
- Compute $e = \mathsf{hash}(C, pk, B, A)$;
- Compute $f = m \cdot e + \beta$ and $w = r \cdot e + \gamma$; and $v = r \cdot (e - f) + \delta$
- Return $\pi_5 := (B, A, f, w, v)$

Verifier:

- Compute $e = \mathsf{hash}(C, pk, B, A)$ and return valid if and only if
  - $C^e \cdot B = \mathsf{EC.Enc}_{\mathsf{pk}}(f; w)$
  - $C^{e-f} \cdot A = \mathsf{EC.Enc}_{\mathsf{pk}}(0; v)$

Figure 4: Non-Interactive Zero Knowledge proof for lifted Elgamal Encryption of $0/1$

A.5 NIZK for Correct Lifted Elgamal Decryption

Non-Interactive Zero Knowledge proof for correct lifted Elgamal decryption

$$\mathsf{NIZK}\{(\mathsf{pk}, C, D), (\mathsf{sk}) : D = C_1^{\mathsf{sk}} \wedge \mathsf{pk} = g^{\mathsf{sk}}\}$$

Statement: Public key, $\mathsf{pk} := h \in \mathbb{G}$, ciphertext $C := (C_1, C_2)$, and the decryption share D
Witness: $\mathsf{sk} \in \mathbb{Z}_p$

Prover:

- Pick random $w \leftarrow \mathbb{Z}_p$; Compute $A_1 := g^w$ and $A_2 := C_1^w$
- Compute $e = \mathsf{hash}(C, D, A_1, A_2)$ and $z = \mathsf{sk} * e + w$
- Return $\pi_4 := (A_1, A_2, D, z)$

Verifier:

- Compute $e = \mathsf{hash}(C, D, A_1, A_2)$; and $D = \frac{C_2}{g^m}$, and return valid if and only if
  - $g^z = h^e \cdot A_1$
  - $C_1^z = D^e \cdot A_2$

Figure 5: Non-Interactive Zero Knowledge proof for correct lifted Elgamal decryption

A.6 NIZK for m

Non-Interactive Zero Knowledge proof for knowledge of message m

$$\mathsf{NIZK}\{(\mathsf{pk}, C), (m, r) : C = \mathsf{EC.Enc}_{\mathsf{pk}}(m; r) \wedge m \in \{0,1\}^{\lambda}\}$$

Statement: Public key, $\mathsf{pk} := h \in \mathbb{G}$, and ciphertext $C := (C_1, C_2) = (g^r, g^m h^r)$
Witness: $m \in \{0,1\}^{\lambda}$ and $r \in \mathbb{Z}_p$

Prover:

- Pick random $s, t \leftarrow \mathbb{Z}_p$; Compute $E = \mathsf{EC.Enc}_{\mathsf{pk}}(s; t)$
- Compute $e = \mathsf{hash}(C, pk, E)$; and $w = e \cdot m + s$; and $v = e \cdot r + t$
- Return $\pi_6 := (w, v, E)$

Verifier:

- Compute $e = \mathsf{hash}(C, pk, E)$ and return valid if and only if
  - $\mathsf{EC.Enc}_{\mathsf{pk}}(w; v) = C^e \cdot E$

Figure 6: Non-Interactive Zero Knowledge proof for knowledge of message m

A.7 Designated NIZK for Message m

Designated Non-Interactive Zero Knowledge proof for knowledge of message m

$$\mathsf{NIZK}\{(\mathsf{pk}, C, \hat{e}), (m, r) : C = \mathsf{EC.Enc}_{\mathsf{pk}}(m; r) \wedge m \in \{0,1\}^{\lambda}\}$$

Statement: Public key, $\mathsf{pk} := h \in \mathbb{G}$, and ciphertext $C := (C_1, C_2) = (g^r, g^m h^r)$ and $\hat{e} = \mathsf{EC.Enc}_{\mathsf{pk}}(e)$
Witness: $m \in \{0,1\}^{\lambda}$ and $r \in \mathbb{Z}_p$

Prover:

- Pick random $s, t \leftarrow \mathbb{Z}_p$; Compute $E = \mathsf{EC.Enc}_{\mathsf{pk}}(s; t)$ and $B = \hat{e}^m \cdot Enc(s)$ and $D = \hat{e}^r \cdot Enc(t)$
- Return $\pi_6' := (B, D, E)$

Verifier:

- Compute $e = \mathsf{Dec}(\hat{e})$ and return valid if and only if
  - $B, D = C^e \cdot E$

Figure 7: Designated Non-Interactive Zero Knowledge proof for knowledge of message m

A.8 Designated Verifier NIZK for π

Designated Verifier Non-Interactive Zero Knowledge proof

$$\mathsf{NIZK}\{(\mathsf{pk}, C, X), (s) : C = \mathsf{EC.Enc}_{\mathsf{pk}}(e) \wedge X = g^s\}$$

Statement: Public key, $\mathsf{pk} := h \in \mathbb{G}$, and ciphertext $C := (C_1, C_2) = (g^r, g^e h^r)$ and X
Witness: $s, r \in \mathbb{Z}_p$

Prover:

- Pick random $w \leftarrow \mathbb{Z}_p$; Compute $A = g^w$; and $D = C^s \cdot \mathsf{EC.Enc}_{\mathsf{pk}}(w)$
- Return $\pi_7 := (A, D)$

Verifier:

- Compute $e = \mathsf{Dec}_{sk}(C)$ and $z = \mathsf{Dec}_{sk}(D)$ and return valid if and only if
  - $g^z = X^e \cdot A$

Randomise:

- Pick random $v, t \leftarrow \mathbb{Z}_p$
- Compute $A' = A \cdot g^v$ and $X' = X \cdot g^t$ and $D' = D \cdot C^t \cdot \mathsf{EC}.Enc_{pk}(v)$
- $\pi'_7 := (A', D')$
- Return $\pi_8 := (X', \pi'_7)$

Figure 8: Designated Verifier Non-Interactive Zero Knowledge proof

B Security Proofs

The security of the treasury voting protocol is analysed in the universal composability (UC) framework. We provide Theorem 3.

Theorem 3. *Let $t, k, n, m = \text{poly}(\lambda)$ and $k/2 < t \leq n$. Protocol $\Pi_{\text{VOTE}}^{t,k,n,m}$ described in Fig. 3.11 UC-realizes $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ in the $\{\mathcal{F}_{\text{LEDGER}}, \mathcal{F}_{\text{DKG}}^{t,k}\}$-hybrid world against static corruption under the DDH assumption.*

Proof. To prove the theorem, we construct a simulator $\mathcal{S}$ such that no non-uniform PPT environment $\mathcal{Z}$ can distinguish between (i) the real execution $\mathsf{EXEC}^{\mathcal{F}_{\text{LEDGER}}, \mathcal{F}_{\text{DKG}}^{t,k}}_{\Pi_{\text{VOTE}}^{t,k,n,m}, \mathcal{A}, \mathcal{Z}}$ where the parties $\mathcal{V} := \{\mathsf{V}_1, \ldots, \mathsf{V}_n\}$, $\mathcal{E} := \{\mathsf{E}_1, \ldots, \mathsf{E}_m\}$ and $\mathcal{C} := \{\mathsf{C}_1, \ldots, \mathsf{C}_k\}$ run protocol $\Pi_{\text{VOTE}}^{t,k,n,m}$ in the $\{\mathcal{F}_{\text{LEDGER}}, \mathcal{F}_{\text{DKG}}^{t,k}\}$-hybrid world and the corrupted parties are controlled by a dummy adversary $\mathcal{A}$ who simply forwards messages from/to $\mathcal{Z}$, and (ii) the ideal execution $\mathsf{EXEC}_{\mathcal{F}_{\text{VOTE}}^{t,k,m,n}, \mathcal{S}, \mathcal{Z}}$ where the parties interact with functionality $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ in the ideal model and corrupted parties are controlled by the simulator $\mathcal{S}$. Let $\mathcal{V}_{\mathsf{cor}} \subseteq \mathcal{V}$, $\mathcal{E}_{\mathsf{cor}} \subseteq \mathcal{E}$ and $\mathcal{C}_{\mathsf{cor}} \subseteq \mathcal{C}$ be the set of corrupted voters, experts and voting committee members, respectively. Note that the underlying encryption scheme is IND-CPA secure, and its corresponding NIZK proofs are simulatable ZK under the DDH assumption.

Simulator. The simulator $\mathcal{S}$ internally runs $\mathcal{A}$, forwarding messages to/from the environment $\mathcal{Z}$. The simulator $\mathcal{S}$ simulates honest voters $\mathsf{V}_i \in \mathcal{V} \setminus \mathcal{V}_{\mathsf{cor}}$, honest experts $\mathsf{E}_i \in \mathcal{E} \setminus \mathcal{E}_{\mathsf{cor}}$, trustees $\mathsf{C}_j \in \mathcal{C} \setminus \mathcal{C}_{\mathsf{cor}}$ and functionalities $\mathcal{F}_{\text{LEDGER}}, \mathcal{F}_{\text{DKG}}^{t,k}$. In addition, the simulator $\mathcal{S}$ simulates the following interactions with $\mathcal{A}$.

In the preparation phase:

- Upon receiving $(\text{INITNOTIFY}, \mathsf{sid}, \mathsf{C}_j)$ from the external $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ for an honest voting committee $\mathsf{C}_j \in \mathcal{C} \setminus \mathcal{C}_{\mathsf{cor}}$, the simulator $\mathcal{S}$ acts as C_j, following the protocol $\Pi_{\text{VOTE}}^{t,k,n,m}$ as if C_j receives $(\text{INIT}, \mathsf{sid})$ from the environment $\mathcal{Z}$.
- $\mathcal{S}$ simulates $\mathcal{F}_{\text{DKG}}^{t,k}$ so that it generates and stores $(\mathsf{pk}, \mathsf{sk})$.

In the voting/delegation phase:

- Upon receiving $(\text{VOTENOTIFY}, \mathsf{sid}, \mathsf{E}_j)$ from the external $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ for an honest expert $\mathsf{E}_j \in \mathcal{E} \setminus \mathcal{E}_{\mathsf{cor}}$, the simulator $\mathcal{S}$ reads pk from $\mathcal{F}_{\text{DKG}}^{t,k}$. $\mathcal{S}$ computes $\mathbf{c_j}^{(3)} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathbf{0}^{(3)})$ and simultates its NIZK proof π_j using the NIZK simulator. It then executes macro $\mathsf{Send\text{-}Msg}\Big((\mathbf{c_j}^{(3)}, \pi_j), \{\mathsf{In}_\eta^{(\mathsf{E}_j)}\}_{\eta=1}^{\ell_1}, \{\mathsf{Out}_\eta^{(\mathsf{E}_j)}\}_{\eta=1}^{\ell_2}\Big)$.
- Upon receiving $(\text{CASTNOTIFY}, \mathsf{sid}, \mathsf{V}_i, \alpha_j)$ from the external $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ for an honest expert $\mathsf{V}_i \in \mathcal{V} \setminus \mathcal{V}_{\mathsf{cor}}$, the simulator $\mathcal{S}$ reads pk from $\mathcal{F}_{\text{DKG}}^{t,k}$. $\mathcal{S}$ computes $\mathbf{u_i}^{(m+3)} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathbf{0}^{(m+3)})$ and simultates its NIZK proof π_i using the NIZK simulator. It then executes macro $\mathsf{Send\text{-}Msg}\Big((\mathbf{u_i}^{(m+3)}, \sigma_i, \alpha_i), \{\mathsf{In}_\eta^{(\mathsf{V}_i)}\}_{\eta=1}^{\ell_1}, \{\mathsf{Out}_\eta^{(\mathsf{V}_i)}\}_{\eta=1}^{\ell_2}\Big)$.
- Once the simulated $\mathcal{F}_{\text{LEDGER}}$ receives $(\text{POST}, \mathsf{sid}, \mathbf{c_j}^{(3)})$ from a corrupted expert $\mathsf{E}_j \in \mathcal{E}_{\mathsf{cor}}$, the simulator $\mathcal{S}$ uses sk to decrypt $\mathbf{c_j}^{(3)}$, obtaining the vote v_j. $\mathcal{S}$ then sends $(\text{VOTE}, \mathsf{sid}, v_i)$ to $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ on behave of E_j.

- Once the simulated $\mathcal{F}_{\text{LEDGER}}$ receives $(\text{POST}, \mathsf{sid}, \mathbf{u_i}^{(m+3)}, \alpha_i)$ from a corrupted voter $\mathsf{V}_i \in \mathcal{V}_{\mathsf{cor}}$, the simulator $\mathcal{S}$ uses sk to decrypt $\mathbf{u_i}^{(m+3)}$, obtaining the vote v_i. $\mathcal{S}$ then sends $(\text{VOTE}, \mathsf{sid}, v_i, \alpha_i)$ to $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ on behave of V_i.

In the tally phase:

- Upon receiving $(\text{DELCALNOTIFY}, \mathsf{sid}, \mathsf{C}_j)$ from the external $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ for an honest trustee $\mathsf{C}_j \in \mathcal{C} \setminus \mathcal{C}_{\mathsf{cor}}$, the simulator $\mathcal{S}$ acts as C_j, following the protocol $\Pi_{\text{VOTE}}^{t,k,n,m}$ as if C_j receives $(\text{DELCAL}, \mathsf{sid})$ from $\mathcal{Z}$.

- Upon receiving $(\text{TALLYNOTIFY}, \mathsf{sid}, \mathsf{C}_j)$ from the external $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$ for an honest trustee $\mathsf{C}_j \in \mathcal{C} \setminus \mathcal{C}_{\mathsf{cor}}$, the simulator $\mathcal{S}$ acts as C_j, following the protocol $\Pi_{\text{VOTE}}^{t,k,n,m}$ as if C_j receives $(\text{TALLY}, \mathsf{sid})$ from $\mathcal{Z}$.

- Upon receiving $(\text{LEAKDEL}, \mathsf{sid}, \delta)$ from the external $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$, the simulator $\mathcal{S}$ simulates the last honest committee C_t's decryption share according to δ, and it simulates the corresponding NIZK proof in the 1st round of the tally phase.

- Upon receiving $(\text{LEAKTALLY}, \mathsf{sid}, \tau)$ from the external $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$, the simulator $\mathcal{S}$ simulates the last honest committee C_t's decryption share according to τ, and it simulates the corresponding NIZK proof in the 2nd round of the tally phase.

Indistinguishability. The indistinguishability is proven through a series of hybrid worlds $\mathcal{H}_0, \ldots, \mathcal{H}_4$.

Hybrid $\mathcal{H}_0$: It is the real protocol execution $\mathsf{EXEC}^{\mathcal{F}_{\text{LEDGER}}, \mathcal{F}_{\text{DKG}}^{t,k}}_{\Pi_{\text{VOTE}}^{t,k,n,m}, \mathcal{A}, \mathcal{Z}}$.

Hybrid $\mathcal{H}_1$: $\mathcal{H}_1$ is the same as $\mathcal{H}_0$ except that during the voting/delegation phase, in $\mathcal{H}_1$, the honest voter V_j's encrypted ballots are replaced with $\mathsf{Enc}_{\mathsf{pk}}(\mathbf{0}^{(m+3)})$ and simulates its corresponding unit vector NIZK proof.

Claim 1. *$\mathcal{H}_1$ and $\mathcal{H}_0$ are indistinguishable if the underlying encryption scheme is IND-CPA and the unit vector NIZK proof is simulatable ZK.*

Proof. The proof is straightforward. Namely, if an adversary $\mathcal{A}$ can distinguish $\mathcal{H}_1$ from $\mathcal{H}_0$, then we can construct an adversary $\mathcal{B}$ who can either break the IND-CPA game of the PKE encryption or the ZK probability of the unit vector NIZK proof. □

Hybrid $\mathcal{H}_2$: $\mathcal{H}_2$ is the same as $\mathcal{H}_1$ except the following: During the voting/delegation phase, in $\mathcal{H}_2$, the honest expert E_i's encrypted ballots are replaced with $\mathsf{Enc}_{\mathsf{pk}}(\mathbf{0}^{(3)})$ and simulates its corresponding unit vector NIZK proof.

Claim 2. *$\mathcal{H}_2$ and $\mathcal{H}_1$ are indistinguishable if the underlying encryption scheme is IND-CPA and the unit vector NIZK proof is simulatable ZK.*

Proof. The same as the previous proof. □

Hybrid $\mathcal{H}_3$: $\mathcal{H}_3$ is the same as $\mathcal{H}_2$ except the followings. During the tally phase, $\mathcal{H}_3$ uses NIZK simulator to simulate all the decryption proofs instead of the real ones.

Claim 3. *$\mathcal{H}_3$ and $\mathcal{H}_2$ are indistinguishable if the underlying decryption NIZK is simulatable ZK.*

Proof. The advantage of the adversary is bounded by the ZK property of the decryption NIZK. □

Hybrid $\mathcal{H}_4$: $\mathcal{H}_4$ is the same as $\mathcal{H}_3$ except the followings. During the tally phase, the honest committee C_t's decryption shares are backwards calculated from the δ and τ received from the $\mathcal{F}_{\text{VOTE}}^{t,k,m,n}$.

Claim 4. *$\mathcal{H}_4$ and $\mathcal{H}_3$ are statistically indistinguishable.*

Proof. The distribution of the decryption shares in $\mathcal{H}_4$ have identical distribution to the shares in $\mathcal{H}_3$. □

The adversary's view of $\mathcal{H}_4$ is identical to the simulated view $\mathsf{EXEC}_{\mathcal{F}_{\text{VOTE}}^{t,k,m,n},\mathcal{S},\mathcal{Z}}$. Therefore, no PPT $\mathcal{Z}$ can distinguish the view of the ideal execution from the view of the real execution with more than negligible probability. This concludes our proof. □