Journal of
*Theoretical and Applied Electronic Commerce Research*

MDPI

*Article*

# Trustful Blockchain-Based Framework for Privacy Enabling Voting in a University

**Vlad Diaconita *, Anda Belciu and Maria Georgiana Stoica**

Department of Economic Informatics and Cybernetics, Bucharest University of Economic Studies, 010374 Bucharest, Romania
* Correspondence: diaconita.vlad@ie.ase.ro

**Abstract:** In this study, we explore the challenges and potential solutions to blockchain-based voting. As a first step, we present a comparison of the relevant platforms for implementing smart contracts in decentralized applications (dApps). We analyze the top platforms, highlighting their advantages and disadvantages, their architecture, and which are more reliable for developing smart contracts. The goal is to find a technology that offers various facilities to the developer and multiple functionalities and performance in the development of smart contracts in a field that has seen an incredible pace of innovation. Based on the findings from our research, we propose a framework based on blockchain technology and smart contracts for university-level voting based on blockchains.

**Keywords:** blockchain; smart contracts; education; voting; Ethereum; Hyperledger; Web3



**Citation:** Diaconita, V.; Belciu, A.; Stoica, M.G. Trustful Blockchain-Based Framework for Privacy Enabling Voting in a University. *J. Theor. Appl. Electron. Commer. Res.* **2023**, *18*, 150–169. https://doi.org/10.3390/jtaer18010008

Academic Editor: Jani Merikivi

Received: 28 November 2022
Revised: 28 December 2022
Accepted: 8 January 2023
Published: 10 January 2023



## 1. Introduction

Cryptocurrencies introduced blockchain (BC) technology to the general public. Although blockchain, which is a decentralized, distributed digital ledger, is separate from cryptocurrencies, they are intertwined in public perception. Trust or lack of trust in cryptocurrencies can attract a similar perception regarding BC technology, and vice versa [1].

The term smart contract (SC) refers to an agreement between participants who do not trust each other. This agreement is automatically enforced by the consensus mechanism of the blockchain, without relying on a third party [2]. The first public BC platform to introduce SC was Ethereum, and Hyperledger was the first enterprise platform to introduced chaincode, a way of implementing SC.

As trust in these technologies is established [3], BC enhanced with SC has the potential to impact various fields such as finance, e-commerce, supply chain management, education, voting, or healthcare [4,5]. Such systems can contribute to improving security [6], transparency, and efficiency in the decentralized context of smart applications (dApps). In education, BC can help improve many administrative tasks in a university, ranging from storing information regarding grades, enrollment, transcripts, and diplomas to implementing transparent voting systems and enabling cryptocurrency payments. The University of Nicosia (UNIC), probably the first institution to use BC in higher education, implemented a system to verify academic certificates using a platform developed by block.co, a spinoff company of UNIC [7].

Typically, dApps are constructed using SC with user interfaces mainly developed in Node.js that can interact with the contracts and BC using their contract application binary interface (ABI). SCs are deterministic and Turing complete. However, some authors argue that non-Turing complete smart contracts could be sufficient, as only about 7% of the analyzed SCs fall into the complexity class of Turing complete functions [8]. A trustworthy dApp should have open-source code (e.g., published on GitHub) and use a public token to run applications [9,10].

SC applications may need a wallet such as Metamask at the user end. They are considered part of Web3, a set of protocols and data structures constructed around distributed

ledgers that have the ability to read and write data from/to the backend and execute SC-based agreements. One of the main problems is that smart contract programming languages have limitations and can produce error-prone SCs that facilitate crypto hacks.

There is research interest in implementing e-voting solutions using BC [11], including by using IoT-embedded devices [12]. Although online and BC voting is still controversial and lacks wide acceptance in high-stakes political elections [13], as the standard and level of testing they need to meet is high [14], it can be good enough for smaller-scale elections, such as those carried out all the time in universities (electing members of faculty councils, university senate, including student representatives, election of doctoral school members, etc.). For such use cases, it is doubtful that most miners or validators of a BC network will collude and temper the votes. Additionally, people involved in the voting process at a university can be considered familiar with new technologies. They can be more easily trained to work with a wallet and a voting token.

In high-stakes political elections or referendums, in-person ballot voting is the preferred choice. It is secure and offers the necessary privacy. Problems might arise during the voting count, if certain committees are compromised, and the ballots could be altered or added. However, large-scale vote tampering is considered difficult because it requires physical access and compromised personnel. There is also a problem with in-person voting for those who live in isolated communities or outside the country. This problem can be solved using mail-in or drop-in ballots, which tends to be controversial, as voting coercion and vote selling are more likely to occur. Nevertheless, most democracies now use mail-in or even all-mail voting (AMV) [15].

In this paper, we seek to prove that a blockchain-based electronic voting solution for smaller-scale elections can be implemented, considering constraints such as requires minimal costs, and guarantees privacy and security without sacrificing transparency. To prove that these requirements can be met, the paper focuses on these research questions:

RQ1: Which are the research trends in blockchain, with an emphasis on smart contracts and blockchain voting?
RQ2: What are the differences between the main smart-contracts-enabled platforms?
RQ3: Which is the blockchain-centered technology stack that provides high performance with zero or very low deployment costs?
RQ4: Are there any other technologies besides BC and SC required to implement an electronic voting system?
RQ5: What is the most practical way for a BC-enabled voting application to ensure that only eligible voters can cast votes and no one, not even the system administrator or database administrator, can generate new votes or alter submitted ballots?

Our technical solution involves using the Solidity programming language and an Ethereum network to develop a voting token that implements the ERC20 interface [16]. For each voting session, an instance of the contract will be deployed. Each token will have a total supply equal to the total number of eligible voters. Votes cannot be added during the contract's lifetime, so additional votes cannot be generated. Voting tokens cannot be transferred between accounts, and no account can have more than one voting token. To ensure voting privacy, we separate and isolate the definition of the voting session from the distribution of Ethereum addresses and voting tokens. The definition of the session includes defining the voting interval, the voting type, and the voting options. During the distribution phase, voters receive a vote token along with the GöETH required to cast their vote.

In the next section, we analyze the current state of the literature regarding BC, SC, and BC-enabled voting. Next, in Section 3, we compare the main BC platforms which offer SC support; in Section 4, we propose a framework for voting in a university that meets the abovementioned research questions. The advantages and disadvantages of the solution are analyzed in Section 5, and the conclusions are drawn in Section 6.

## 2. Literature Review

### 2.1. Blockchain in WoS Publications

For the study of the specialized literature and to address RQ1, we used an approach inspired by the Prisma guide [17]. We selected papers containing the keyword "Blockchain" from the Web of Science Core Collection (WoS). Only articles published in journals, conference volumes, or specialized books were considered (search date: 7 November 2022). The number of articles per year and the number of articles classified by WoS as highly cited or hot papers, as well as the number of their citations, are presented in Table 1.

**Table 1.** WoS publications which have the blockchain keyword (topic).

| Year | Total Publications | Highly Cited + Hot Papers | Citation Numbers (The Year Refers to the Cited Article) for Highly Cited + Hot Papers |
|---|---|---|---|
| 2013 | 2 | - | - |
| 2014 | 10 | - | - |
| 2015 | 24 | - | - |
| 2016 | 123 | 5 | 4236 |
| 2017 | 667 | 16 | 5316 |
| 2018 | 2337 | 51 | 16,324 |
| 2019 | 4304 | 106 | 21,877 |
| 2020 | 5434 | 124 | 17,316 |
| 2021 | 6198 | 114 | 7348 |
| 2022 | 5189 | 59 | 1689 |
| 2023 | 61 | - | - |
| Total | 24,349 | 475 | 74,106 |

In the blockchain field, we can see a great deal of interest, which leads to many publications. We sorted publications based on citations (Table 2) and included the IF and AIS of the journal, where it exists. It is evident that a high number of citations is not necessarily related to a journal's impact factor. We also notice that the most frequent keyword is IoT (Internet of Things).

**Table 2.** Top papers according to the number of citations.

| Pos | Article | Keywords | Journal IF/AIS (2021) | Citations |
|---|---|---|---|---|
| 1 | Blockchains and smart contracts for the Internet of Things (2016) [18] | Blockchain; distributed systems; internet of things. | 3.476/ 0.613 | 1866 |
| 2 | Industry 4.0: state of the art and future trends (2018) [19] | Industry 4.0; Made-in-China 2025; cyber-physical systems; IoT; cloud computing; blockchain; manufacturing; industrial integration; industrial information integration; interoperability; enterprise architecture; SOA; emerging technology. | 9.018/ 1.103 | 1100 |
| 3 | Blockchain challenges and opportunities: a survey (2018) [20] | Blockchain; consensus algorithms; cryptocurrency; internet of things; smart contract. | 0.825/ 0.603 | 1073 |
| 4 | IoT security: Review, blockchain solutions, and open challenges (2018) [21] | IoT security; blockchain; IoT protocols; network security; data security. | 7.307/ 1.086 | 931 |

**Table 2.** *Cont.*

| Pos | Article | Keywords | Journal IF/AIS (2021) | Citations |
|---|---|---|---|---|
| 5 | Blockchain technology and its relationships to sustainable supply chain management (2019) [22] | blockchain technology; supply chain management; sustainability; barriers; research agenda. | 9.018/ 1.103 | 872 |
| 6 | Where Is Current Research on Blockchain Technology? -A Systematic Review (2016) [23] | - | 3.752/ 0.974 | 765 |
| 7 | Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies (2016) [24] | Altcoins; Bitcoin; blockchain; cryptocurrencies; digital currencies; distributed consensus; survey tutorial. | 33.84/ 7.32 | 760 |
| 8 | Blockchain technology in the energy sector: A systematic review of challenges and opportunities (2019) [25] | Blockchain; distributed ledger; energy decentralization; peer-to-peer energy trading; prosumer; renewable energy. | 16.799/ 2.693 | 716 |
| 9 | The truth about Blockchain (2017) [26] | - | 12.129/ 4.443 | 691 |
| 10 | Designing microgrid energy markets A case study: The Brooklyn Microgrid (2018) [27] | Microgrid energy market; market design; blockchain; case study; Peer-to-peer trading; renewable energy. | 11.446/ 1.87 | 685 |
| | | [ . . . ] | | |
| 197 | A Smart Contract for Boardroom Voting with Maximum Voter Privacy (2017) [11] | - | Proceeding paper | 181 |
| | | [ . . . ] | | |
| 385 | Blockchain-Enabled E-Voting [28] | Blockchain-enabled e-voting; BEV; e-voting; blockchains; elections; voter fraud; voter access; paper ballots; electronic voting; online voting; software development; software engineering. | 3/ 0.997 | 119 |

On the first position, there is a paper from 2016 that presents blockchains and smart contracts for IOT and has almost 70% more citations than the paper in the next position, which is about Industry 4.0. That is because IOT has been a well-discussed topic over the past decade.

It is not surprising that reviews, surveys, and other articles that discuss BC in a more general context are the most cited. Furthermore, the Internet of Things (IoT) seems to be a topic that generates many citations. The first article in the top that discusses smart contracts is at position 3, and the first that focuses on voting using smart contracts for e-voting is at 197, well outside the top 10; it was published in a conference proceeding and cited 181 times. Furthermore, recent articles [12,29–31] also deal with electronic voting using BC and focus on privacy, so we can conclude that it is a rather hot topic within the BC ecosystem.

### *2.2. Blockchain Voting*

From the classical voting system to the blockchain voting, several stages have been reached: usage of dedicated voting machines, optical scanning of votes, electronic ballot printers, and software for voting through the internet [32].

Blockchains that are used for e-voting should ensure transparency, anonymity, auditability, dependability, consistency, public and individual verifiability [33], fairness, data integrity, robustness, and uniqueness [34]. In such a context, it is very important not to allow the users to vote more than once, not to see intermediate results which could affect the vote of others, and not to reveal the voter's identity or their voting preferences. The blockchain should not be attackable and should accept the same outcome of the election.

Some blockchain voting systems do not allow users to change their vote in case of an error [35,36], while others allow a change in a time limit set by an election committee [37,38].

In Table 3, there are four e-voting platforms presented, considering the blockchain they use, the development environment, and the smart contracts.

**Table 3.** E-voting projects.

| E-Voting Blockchain Projects | Blockchain | Development | Smart Contract |
|---|---|---|---|
| Public Votes | Ethereum | Meteor | 1 coded in Solidity |
| Luxoft | Hyperledger Fabric | Hyperledger Fabric | Yes |
| Ethereum Blockchain Trustless Voting | Ethereum | Python, Javascript | Many in Solidity |
| Follow My Vote | BitShares | C++ | Pollaris |

For PublicVotes, the user that creates the poll pays for the creation of the poll and for all votes. The solution of Luxoft has an innovative encryption technology that anonymizes the votes and allows a secure audit. It uses Amazon AWS, the Lucerne University of Applied Sciences' data center and n'cloud.swiss, so that the main platform is deployed on three different data centers in the cloud. Ethereum Blockchain Trustless Voting is an open-sourced voting system, existing as a smart contract running on Ethereum that uses threshold keys and linkable ring signatures [33].

For Follow My Vote, the voter downloads and installs the Voting Booth. After the user is authorized, he can vote and even change his vote, if the election officials allow it. The voter can also audit each ballot to confirm that the election results are accurate.

Because the blockchain technology was intensively used over the past decade, it has reached its maturity and thus can be trustfully used for sensitive domains such as e-voting, but with extra cautions regarding anonymity, authentication, and end-to-end verifiability. In a research conducted over 437 papers, it was emphasized that Ethereum is the main development platform (19 out of 52 research papers) chosen for voting systems [32].

## 3. The Main Smart Contract Platforms

Smart contract platforms can be categorized into public platforms, anonymous public platforms, and enterprise (or private) platforms, based on the level of permission requirements to access the platform. To answer RQ2, we analyze these platforms in this section to determine which is most suitable for privacy-enhanced voting.

### *3.1. Public Platforms*

Choosing a blockchain network and determining when to use the different BC technologies, projects, and protocols are common issues as more and more developers turn to the BC ecosystem. Addressing these issues requires a thorough understanding of the differences between these technologies. The main public platforms employ layer 1 (the base BC) and layer 2 solutions and projects (networks built on top of layer 1). Data stored on layer 1 are considered "on-chain", whereas layer 2 facilitates "off-chain" transactions. Layer 2 solutions are usually built to improve scalability and reduce costs.

Table 4 illustrates the significant differences between the most popular blockchain platforms for smart contract development. We will discuss public networks in this section and enterprise networks in the next section.

As can be seen in Table 4, Ethereum and Hyperledger are the platforms that also support B2B applications, while the other ones only allow B2C applications. For the consensus algorithm, most of the platforms use Proof of Stake (PoS), while Internet Computer uses the Threshold Relay technique and Hyperledger uses a Pluggable consensus mechanism. By far, the fastest blockchains are Solana and Polygon, with an average of 65,000 transactions per second. Ethereum and Internet Computer have a stateful architecture, while Solana and Hyperledger have a stateless one, and Polygon uses a multichain architecture.

**Table 4.** The differences between the main SC-enabled platforms.

| Criteria | Ethereum | Solana | Polygon | Internet Computer | Hyperledger |
|---|---|---|---|---|---|
| Cryptocurrency | ETH | SOL | MATIC | ICP | None |
| Confidentiality | Public | Public | Public | Public | Private |
| Purpose | Mainly B2C applications, but also supports B2B applications | B2C | Faster B2C over Ethereum | B2C, front-end + back-end | Mainly enterprise-level B2B applications, but also supports B2C |
| Programming languages | Solidity, Vyper, Yul | Rust, C/C++ | Solidity, Vyper | Motoko | Go, JavaScript, TypeScript, Java |
| Consensus algorithm | PoS (as of 15 September 2022) | PoS, PoH | PoS, Plasma-based sidechain | Threshold Relay | Pluggable consensus mechanism (e.g., pBFT, round-robin, PoW) |
| Who pays the SC fees? | The user | The user | The user | Usually, the canister (SC) | No fees |
| Transaction average speed | 13–14/s (PoW) → 12/s (PoS) | 50,000–65,000/s | 65,000/s | 11,500/s | 3000–20,000/s |
| Architecture | Stateful | Stateless | Multichain | Stateful | Stateless |
| Readily available test networks | Yes, multiple | No | No | No | No |
| Scalability | Limited | High-performance protocol for scalability | Average | Unlimited | Yes |
| First appeared | 2013 | 2017 | 2017 | 2016 | 2016 |
| Headquarters | Bern, Switzerland | San Francisco, California, United States | Bengaluru, Karnataka, India | Zürich, Switzerland | San Francisco, California, United States |

In terms of scalability (i.e., how well the platform can adapt to new flows of transactions), Solana is one of the most scalable blockchains, based on its high-performance protocol for scalability. Blockchain scalability can be vertical and horizontal [39]. Horizontal scalability needs to prove the elasticity of the system when adding new resources, such as machines or servers, whenever it is demanded. This can be performed by adding new nodes and clients without affecting the performance of the blockchain. Vertical scalability assumes improving the existing nodes, so the transactions are processed in a more efficient way. It can be performed by adjusting some elements such as block size, sharding, lightening, or parallel mining.

In decentralized financing applications and new financial technologies, **Ethereum** was the first, and remains the most well-known, blockchain platform. From ICOs to smart contracts, it can facilitate the implementation of nearly any kind of decentralized application. The total market cap of ETH, the cryptocurrency of Ethereum, is more than 4.5 times larger than BNB, the next SC-enabled BC, and more than 10 times larger than ADA (10 October 2022). Although the term smart contracts predates BC [40], today it is related to it, as it differentiates Blockchain 1.0 platforms from Blockchain 2.0 ones.

Blockchains can be classified into four stages based on the evolution of technology [41]. The original Blockchain 1.0 (e.g., Bitcoin, Ripple, Dash) was dedicated to cryptocurrency and distributed ledger for storing and transferring value, while Blockchain 2.0 (e.g., Ethereum) focused on smart contracts and distributed and decentralized applications. With Blockchain 3.0 (e.g., Hyperledger, R3 Corda), many individuals could reach this technology because

more industries, such as healthcare, education, or e-Commerce, were influenced by enterprise blockchains. Finally, Blockchain 4.0 (e.g., RChain) refers to the industry-infrastructure-based blockchain ecosystem.

All decentralized applications in Ethereum are run by an Ethereum Virtual Machine (EVM), which executes code of different complexity. Smart contracts are stored on the Ethereum blockchain.

As an architecture, Ethereum offers a stateful design, meaning that it records all transactions in their current, existing state. After "the merge" (i.e., the transition from Proof-to-Stake (PoW) to Proof-of-Stake (PoS)) was finalized on 15 September 2022, there was 99% less energy consumption for ETH, as expected [42]. This solved a major problem, as in PoW, one transaction used 200.05 kWh and the entire Ethereum network accounted for about 0.2% of the worldwide electricity consumption [43]. As PoS made mining absolute, ETH miners had to shut down or migrate to remaining PoW coins such as ETC, BEAM, or RVN, resulting in a hash rate increase and a steep miner's fee decrease for those coins. Ethereum scalability [44] should eventually increase to 100,000 transactions per second from 13–15 processes per second, as was the case when PoW was used.

Although Ethereum was criticized for its slow trading speed and high fees, it remained the number one SC chain due to its first-mover advantage. As a result of the PoS consensus algorithm, Ethereum is likely to retain its status as the most influential blockchain platform for smart contracts.

The **Solana** platform aims to improve its attractiveness by providing one of the fastest BCs, with an average of 50,000–65,000 transactions per second [45]. It uses a stateless architecture for SC, so previous transactions are neither stored nor referenced. In addition to PoS, Solana introduced a new Proof-of-History (PoH) consensus mechanism. The consensus approach uses SHA-256 hashing to verify blockchain transactions through multiple nodes on the network. Solana claims to be scalable, decentralized, and secure, effectively solving the "scalability trilemma" (at one time, we can only have two of them: decentralization, scalability, and security [46])) associated with previous blockchains, such as Ethereum [47]. In addition to being environmentally friendly and energy-efficient, BC is also more sustainable. It is possible to register aliases for account addresses and tokens through SC to reduce user input error [48]. In addition, multiple programming languages are available this time, including Rust and C/C++ for smart contracts. Since there are currently not many validators, Solana is somewhat vulnerable to centralization. Anyone can become a Solana validator, which can be expensive on a large scale, since it requires a lot of resources.

In Figure 1, the same withdraw function is written in Rust for Solana and in Solidity for Ethereum. Because of the stateless design, in Solana, all states must be provided as input parameters; thus, such smart contracts are more complicated to write and understand.

A plasma chain parallel to Ethereum is **Polygon**, formerly known as Matic, which uses the Ethereum blockchain but aims to improve performance and transaction costs. Polygon performs even better than Solana in terms of speed and performance: it processes an average of 65,000 transactions per second [45], with fees that cost less than a fraction of a cent, and completes the process of confirming the transaction in a single block. With an exemplary track record in terms of security, performance, and speed, Polygon plans to expand beyond Ethereum and become a leader in BC video game solutions and NTFs. It differentiates itself by adopting a different scaling technology, that is, an adapted plasma-based version. Solidity and Vyper are used as development technologies for smart contracts with Polygon. A disadvantage of using the Polygon blockchain is that it is not standalone. Because Polygon works over Ethereum, if Ethereum ceases to exist, Polygon will as well.

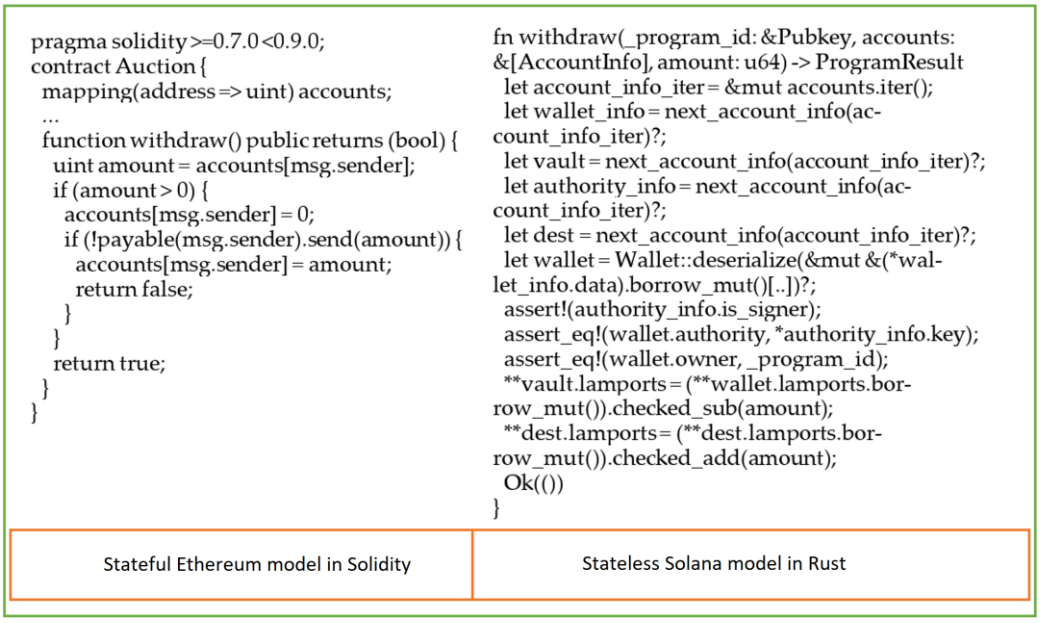


**Figure 1.** Ethereum and Solana programming models [49].

**The Internet Computer** introduces the concept of canisters that are basically stateful SCs. It scales by using an open algorithmic governing system that can add, when needed, extra subnets and nodes [50]. It can currently process up to 11,500 transactions/second and 250,000 queries per second.

The storage of data on certain platforms tends to be prohibitively expensive. Just keeping 100 kilobytes on the Ethereum Mainnet costs over 3 ETH, or USD 3500 at early November 2022 prices. For this reason, it makes sense to use decentralized storage solutions such as the Interplanetary File System (IPFS), which provides off-chain decentralized storage for large files and communications at much lower costs, especially compared to the Mainnet [51]. The solutions in this category can be considered content delivery networks using BitTorrent-like protocols. IPFS currently has a total network storage capacity of more than 19 EiB (https://filfox.info/en (retrieved 9 November 2022)). Tokens are frequently used to incentivize network players in Web3. Tokens are similar to the rewards and loyalty programs [52] of many large companies (e.g., you earn "miles" when you purchase with an affiliated credit card). In IPFS, for example, FileCoin tokens are used to repay storage providers and retrieval providers. In the chain, these deals are published as contracts. In the chain, client funds are locked for the duration of the transaction, after which they are released to providers. A FileCoin token can also be traded on both centralized and decentralized markets in exchange for other cryptocurrencies or fiat currencies.

In addition to IPFS, other off-chain solutions can be used in relation to a BC. There are the common layer 2 protocols: nested BC, sidechains, state channels, and rollups. Beyond these protocols, other solutions, such as Apache Kafka, can make good use of the different APIs for providing integration and interoperability between BC and other systems of an organization (business applications, databases, cloud storage, etc.).

Many blockchain platforms are public, so anyone can view transactions through dedicated websites such as www.blockchain.com/explorer (accessed on 9 November 2022) for Bitcoin or https://etherscan.io (accessed on 9 November 2022) for Ethereum. A platform of this kind allows us to see the metadata of each block, such as block height, timestamp, block reward, difficulty, hash, nonce, and gas used for Ethereum, along with a list of transactions for that block, including the "From" and "To" addresses, and their value in Ethereum.

### 3.2. Public Anonymous Platforms

There are also BC networks such as Monero or Zcash that only publish limited data regarding transactions on their block explorer, shielding certain information such as the sender, the receiver, and the amount. These networks have limited SC support. Monero uses the Ring Confidential Transaction Protocol (RingCT), for which different improvements have been proposed, such as Aggregation Ring Confidential Transaction (ARCT) or RingCT 3.0 [53,54]. Zcash uses a different mechanism, Zerocash, based on zero-knowledge proof constructions (zk-SNARK). These mechanisms are 100% foolproof; there are several papers that examine the deanonymization of Monero and Zcash [55,56]. Furthermore, anonymity makes private data sharing transactions prone to disputes [57], which can be a problem in electronic voting.

**Zcash** (ZEC) enables four types of transactions: t-to-t, t-to-z, z-to-z, and z-to-t [56], where t are visible addresses, z are hidden addresses, and the transactions between hidden addresses form the shielded pool. Nevertheless, it was demonstrated that most of the users do not care so much about anonymity, which is the key feature of Zcash [56]. Newer sources identify the fifth type of transaction, in which z encrypted addresses are involved as senders and receivers, but there also exist public inputs or outputs. Therefore, the fifth transaction type is called mixed [58].

**Monero** uses ring signatures to make the inputs of a transaction private. Any third party can verify the veracity of the signature, but cannot identify the sender [58]. In the Monero platform, every user is anonymous, because it uses these three main techniques: stealth address (through which only the sender and receiver can determine where a payment was sent), ring signatures (which ensure that transaction outputs are untraceable), and RingCT (which facilitates hiding transaction amounts) [59].

Monero and Zcash require more resources than Bitcoin, because for Monero, transactions store 10 decoys for each input; therefore, the transaction increases in size on disk, while Zcash transaction validation requires a lot of memory and time [58]. The main difference between Zcash and Monero is that Zcash uses optional shielding, while Monero uses shields for all transactions [59].

As shown in Table 5, both public blockchains, Monero and Zcash, use PoW algorithms (RandomX, Equihash) and ensure great confidentiality (Ring Confidential Transactions, zk-SNARKs), but are not as good in terms of scalability and auditability. The transaction average speed is twice higher for Monero.

**Table 5.** The differences between Monero and Zcash platforms.

| Criteria | Monero | Zcash |
|---|---|---|
| Cryptocurrency | XMR | ZEC |
| Programming languages | C/C++ | Rust |
| Consensus algorithm | PoW (RandomX) | PoW (Equihash) |
| Transaction average speed | 2/min | 75/s |
| Scalability | Poor | Average |
| Confidentiality | Very good (Ring Confidential Transactions and Stealth Addresses) | Excellent: zk-SNARKs (an advanced form of zero-knowledge cryptography) |
| Auditability | Poor (ViewKey and Payment Proofs) | Poor (Viewing Keys) |
| First appeared | 2014 | 2014 |
| Headquarters | Sydney, New South Wales, Australia | Colorado, United States |

### 3.3. Private Enterprise Platforms

**Hyperledger** was developed to accelerate the development of blockchain technologies that span industries. It is an open-source global collaboration that helps to create and

develop distributed enterprise-grade ledger frameworks. In this framework, numerous libraries, tools, and modules are included. In late 2015, it became the second implementation of Blockchain 2.0 after Ethereum. There are, of course, other enterprise-oriented networks such as R3 Corda or Enterprise Ethereum Alliance (EEA). The latter is focused on facilitating enterprise implementations on the Ethereum BC.

Any business must ensure confidentiality and security, as data breaches can seriously impede organizational development. The transactions taking place through the Ethereum Mainnet can be seen by anyone using it. However, given Hyperledger's strong encryption, only those with proper access can see the transactions. Hyperledger offers a safe method of information transmission between parties.

**Ethereum** is a blockchain network that does not require permission to access data. It is a public ledger, so anyone can download all transactions using the Ethereum client. In other words, anyone with Internet access can participate and become a node. Of course, personalized and private Ethereum deployments for enterprises are possible.

Hyperledger is a BC network with controlled and auditable access moderated by a membership service provider (MSP). Furthermore, certificates are needed to sign every operation. These certificates are usually generated using the Hyperledger Fabric Certificate Authority (CA). In other words, the network is restricted to a set of authorized members, and restrictions can be based on complex business regulations. Roles can be created and assigned to users in a way similar to traditional databases.

Each platform is intended for a different purpose, which is another key difference between Hyperledger and Ethereum. Ethereum is used for decentralized applications that are intended for widespread consumption, whereas Hyperledger is used to run smart contracts in a business-to-business (B2B) environment. Thus, blockchain applications can be customized with restricted access to meet their unique requirements. EEA also makes it possible to use dApps or decentralized applications in a B2B environment. Of course, there are several products (e.g., Hyperledger Cactus, Hyperledger FireFly) that can provide interoperability between different BC networks and support for standards such as ERC20, ERC721, and ERC1155, which can model different assets.

The Ethereum blockchain community shares decision-making to develop and offer support for the platform. The decision-making process will also be open to other stakeholders, such as crypto exchanges, miners, and dApp developers.

The Linux Foundation developed the Hyperledger Fabric. Many companies such as IBM made significant contributions to this framework. The decision-making for each BC implementation varies and is usually constrained by business rules. Hyperledger Fabric is a permissioned private network, which means that each member of the network is known. Due to this, as compared to a public BC such as Ethereum, it is the preferred option for businesses that want to build smart contracts but must adhere to data protection requirements. Due to its private design, Hyperledger Fabric offers adaptability, versatility, complexity, and secrecy of transactions [60].

The main ***programming language*** for Ethereum is Solidity. Although there have been initiatives to develop alternatives such as Vyper or Yul, these have so far failed to stand out and attract a significant market share. These languages, even though they share similarities with widely known programming languages such as C++ or Python, are specific to Ethereum and cannot be used to develop applications in other environments. In Hyperledger Fabric, by importing the necessary modules, programmers can easily write programs (chaincode) in a variety of widely used programming languages (Go, JavaScript, Java). The chaincode is similar to a smart contract but is more appropriate for a whole range of real-world applications where defining roles, assets, and constraints is essential. There were even attempts to introduce a permissioned Ethereum smart contract blockchain node into the Hyperledger ecosystem (e.g., Hyperledger Burrow, which was discontinued in May 2022).

The ***consensus mechanism*** is used to agree on the next block to be added to the chain and has a great impact on the ***speed of the transaction***. Application developers who need

the assurance that their contracts will always be completed fast may be concerned by the network's record of routinely operating at full capacity. Verifications are conducted on the order of operations and the correctness of the transaction. Ethereum started with a PoW consensus algorithm which is very secure but leads to slow transactions. On 15 September 2022, Ethereum switched to another consensus mechanism (i.e., PoS), finalizing "the merge". The change increased the transaction speed, but only barely at the beginning. In PoW, a block was mined every 13–15 s, whereas in PoS a block is now mined every 12 s. Further post-merge planned developments, such as sharding, may further increase transaction speed up to a theoretical potential of 100,000 transactions/s. Hyperledger supports many consensus mechanisms and can process up to 20,000 transactions/s [58]. Choosing one or the other depends on the use case. The most used one is pBFT (practical Byzantine fault-tolerant) where the request is sent to multiple nodes and is accepted if m+1 replies are received from the nodes of the network (m is the maximum accepted number of faulty nodes). Although it is considerably faster than PoW, the problem with pBFT is that it does not scale well [61]. Thus, for secure channels, a simpler round-robin model can be used. Hyperledger can even use PoW if it is used in a public B2C scenario.

In public BC networks, validators obtain an ***incentive*** in the network's currency if they sign a block. Currently, the reward for signing an Ethereum block is about 2 ETH. In Hyperledger, there is no reward. This is a minus, as external organizations might see joining the BC as an added burden regarding cost and risk. Incentivization could be performed by financial modeling and convincing them that this is a better way to run business processes.

Hyperledger differentiates itself by using, in addition to the BC, a so-called world state database to hold the latest values of the attributes. The world state database is pluggable, and multiple solutions such as LevelDB or CouchDB can be used. This is useful in a business environment, as databases excel in storing and retrieving data.

As for the ***number of blockchains***, Ethereum (as most public BCs) has one Mainnet, where the real ETH is used, and multiple test nets (e.g., Goerli, Sepolia) used mainly by developers and powered by worthless ETH. The Goerli Ethereum network, an Ethereum Mainnet fork, is a test net started by the Ethereum team. Together with Sepolia, it replaces previous test nets (Kiln, Ropsten, and Rinkeby) which will be phased out as Ethereum migrates to PoS. Before releasing dApps to the actual Ethereum Mainnet, Goerli enables developers to use a test environment very similar to it. The gas needed to run the SC can be paid for using free ETH (named GöETH) that can be requested from a Goerli faucet. The SC can run indefinitely on Goerli or it can be migrated on the Ethereum Mainnet, which is more secure, but in that situation, the gas must be paid using real ETH.

In Hyperledger Fabric there can be multiple private ledgers (named channels) which are somewhat similar to Apache Kafka's topics, as organizations can belong ("subscribe") to different channels. Furthermore, in the past, Fabric 1.0 used Kafka to ensure crash-fault-tolerant consensus [62].

In light of the features analyzed in this section, as well as the pros and cons of each option, we chose Ethereum as the BC platform for developing the voting solution, as it is the only one that offers a free solution for nonfinancial applications, allowing users to vote without paying in real ETH (the "test" networks such as Goerli). Consequently, we chose Solidity as the smart contracts object-oriented programming language. The anonymous public platforms were not considered, as their anonymity makes private data-sharing transactions vulnerable to disputes [57].

## 4. Methods and Proposed Solution

In this section, we focus on proposing a voting framework for universities. First, a voting token based on the ERC20 interface is developed. An instance of the contract will be deployed for each voting session. The token can have a unique symbol for all voting sessions (e.g., "ASE") or can be personalized for each voting session (e.g., "VOTE15SEP22"). The total supply of each deployed token should be equal to the total number of eligible voters (e.g., 200). When the token is deployed on Goerli, anyone can see the total supply

on https://goerli.etherscan.io/ (Figure 2). This total supply cannot be altered during the lifetime of the contract, so it is not possible to generate additional votes. The token is constructed so that no account can have more than one voting token and tokens cannot be transferred between voters (i.e., the transfer function has require (msg.sender==admin && voteState==State. beforeStart, "Only the admin can transfer tokens, only before the voting has started")) or even by the admin after the voting started.

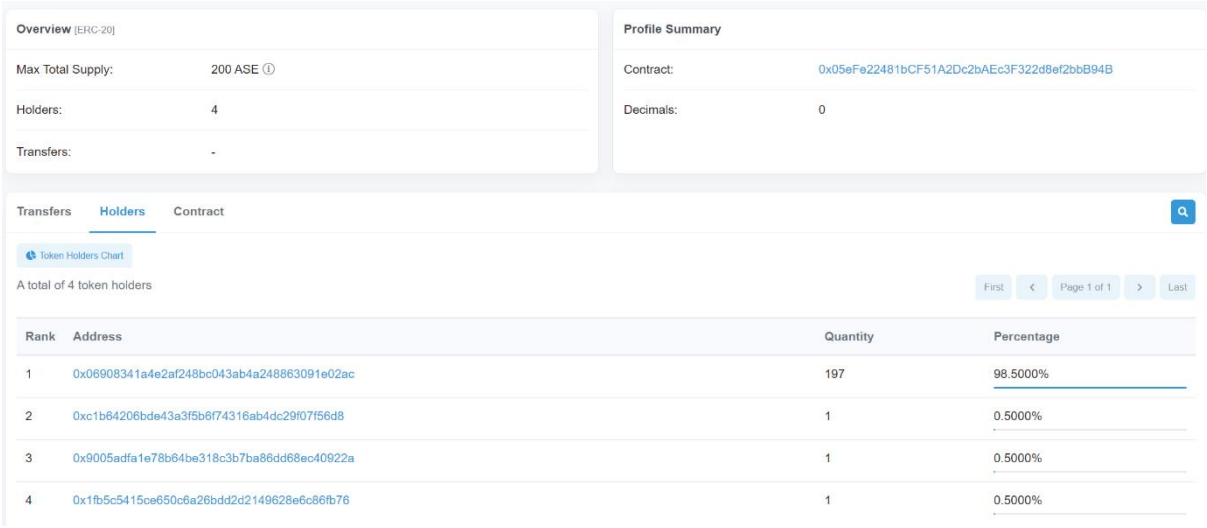


**Figure 2.** The token's etherscan page after the distribution of the voting rights has begun.

The next step is to distribute the tokens to the voters. There could be questions raised if a log is kept that links externally owned accounts (EOAs) to a specific person's name. As shown in the use case diagram (Figure 3), our approach is to separate the definition of the voting session from the distribution of Ethereum addresses and voting tokens. This will allow the distributor to know who he sent each address to, but not how the votes were cast. Voting committees will know how each address voted but not who it belongs to.

In the web application, the voting committee's administrator defines a voting session specifying the voting interval, the voting type, and the voting options. After entering all these data, the voting token is deployed. For example, if the voting is for the business department and there are 200 eligible voters in that department, a voting token with a maximum supply of 200 will be created.

After the token has been deployed, the entire supply belongs to the contract owner. The voting committee will then transfer the contract ownership and a list of eligible voters to the distributor, who can be a person or an automated process. One approach is to generate Ethereum accounts (e.g., by using geth (https://geth.ethereum.org/docs/interface/managing-your-accounts)) for each person and transfer one token to each of those accounts together with the GöETH needed to cast the vote (i.e., to pay for the gas fees). The private keys or the JSON files together with the passwords must be sent to the voters using email or through another channel (e.g., a phone app). A certain trust in the system is needed at this phase but this is common in all BC applications, including cryptocurrency payments. When a wallet is installed and an account generated, the user must presume that the software will not store the secret phrase, as this will give access to the crypto assets associated with that wallet.

Appendix A shows an example of a JSON file containing an Ethereum address. The file and the password can be sent through different channels (e.g., the file by email and the password by a mobile app or by SMS). The voters will first import the account from the JSON file and then add the token as a new asset (Figure 4).


**Figure 3.** Use case diagram.

Another approach would be to transfer the tokens to a user address not automatically, but at the users' request. The request must be made on an internal web page after authorization with an IDM account. Here, requests are recorded so that a person cannot request a token more than once, but not the link between the identity management account and the Ethereum account to guarantee voting anonymity.

On the token etherscan page, anybody can see that the distribution of tokens has begun but does not know to whom those hexadecimal addresses belong to. Figure 2 presents the situation when the distributor sends three tokens to those addresses. After all the tokens have been distributed, the owner of the SC must have zero tokens.

After all tokens have been sent and the voting session has begun, voters can open the voting page, choose the candidate(s) or the option they vote for, and confirm their vote using the token. After the vote, the token is deducted from the balance of the voter, so no one can vote twice. After the token is deducted, it can be sent to an Ethereum address for which nobody has the private key (e.g., the address of the Ethereum genesis block, sometimes called the null address) or it can simply not be added anywhere else. In both cases, the transaction will be visible on etherscan.

As depicted in the sequence diagram (Figure 5), tamperproof data should be stored on-chain (i.e., the token, the votes) while for data that can be changed, it makes better

sense to store them off-chain (i.e., data about the voters and about each voting session, multimedia files). Storing data off-chain makes updating and retrieving easier, reducing costs at the same time if the fees are paid in real ETH. However, after the token SC has been deployed for a voting session, updating off-chain data such as voting interval, voting type, or voting options is no longer possible.

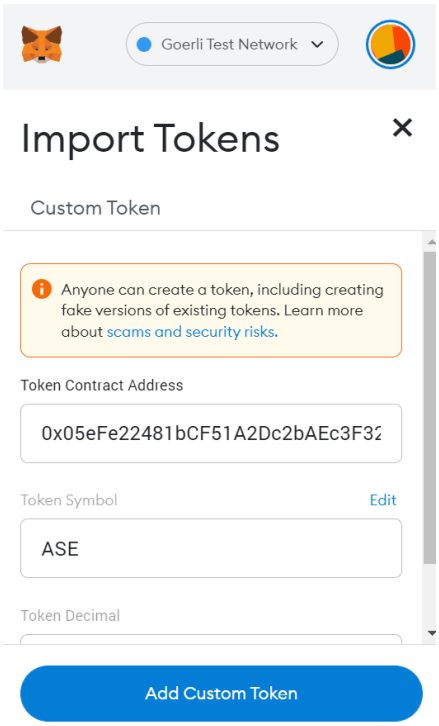


**Figure 4.** Importing the voting token as a custom asset in the Metamask wallet.

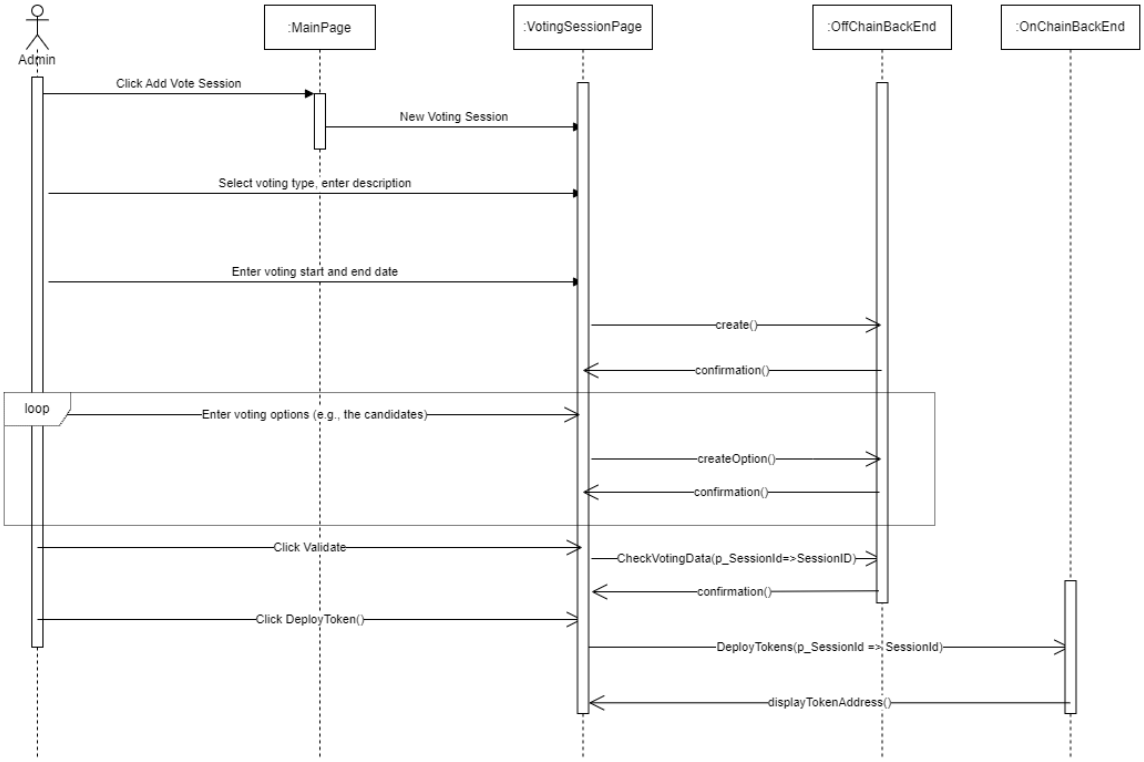


**Figure 5.** Sequence diagram to define a voting session.

## 5. Analysis and Discussion

In this section, we summarize the findings in the context of the research questions.

*RQ1: Which are the research trends in blockchain, with an emphasis on smart contracts and blockchain voting?*

Blockchain and smart contracts are thriving research topics. As presented in Tables 1 and 2, the most cited BC-related research focuses on IoT, privacy, and security, and article types, reviews, and surveys are the most popular, followed by research articles and conference papers. Very recent articles also focus on privacy and security [63] but also on optimizing consensus models [64–67], blockchain-based voting [32], using reinforcement learning to improve external data access without compromising the BC integrity [68], BC-based trading and auctioning [69,70], and supply chain management in the context of cross-border e-commerce [71].

*RQ2: What are the differences between the main smart-contracts-enabled platforms?*

This research question was addressed in length in Section 3. To summarize, each platform type has its specific use cases; there is no one-size-fits all SC platform. Ethereum has been around for a long time, has a large developer community, and has a wide range of application scenarios. In contrast, the Internet Computer and Solana are newer platforms that emphasize high performance and scaling.

*RQ3: Which is the blockchain-centered technology stack that provides high performance with zero or very low deployment costs?*

For developing the voting solution, we chose Ethereum as the BC platform, as it is the only one that provides a free solution for nonfinancial applications (the "test" networks, such as Goerli) and Solidity as the object-oriented programming language. The **advantages** of the approach consist of the following:

- ***No running costs***—the gas fees are paid in GöETH which do not cost real money and can be secured through different faucets. If the Mainnet had been used, voters would have needed to pay the gas fees in real ETH.
- ***Good security***—the threat of a 51% attack is low. For such an attack, someone would require someone to secure control over 51% of the staked GöETH. Even if it is worthless, such an amount is extremely hard to gather through the Goerli faucets. If the Mainnet had been used, an attacker would need 51% of the staked ETH (about USD 15 billion).

*RQ4: Are there any other technologies besides BC and SC required to implement an electronic voting system?*

Yes, as discussed in Sections 3 and 4, other technologies are needed. These include cryptographic solutions for anonymous authentication or secret sharing schemes [72], off-chain solutions to improve scalability, to store data at lower costs, or oracles to access data and trigger actions outside the blockchain [68].

*RQ5: What is the most practical way for a BC-enabled voting application to ensure that only eligible voters can cast votes and no one, not even the system administrator or database administrator, can generate new votes or alter submitted ballots?*

Our voting token approach was presented in Section 4 and has several advantages regarding transparency and privacy.

- **Transparency:**
- The source of the SC is posted together with the compiled form. If the two forms match, the SC will appear as verified on the blockchain, giving users the opportunity to audit the code to make sure it does what is supposed to do (e.g., the votes are recorded and counted correctly, and the token is burnt after the vote).
- The Max Total Supply is visible to anyone so there cannot be more tokens than voters. The holders' addresses (the Keccak-256 hash of the public key of the account) are also visible, so anyone can check if anyone holds more than one token (Figure 2).

- Privacy:
- A separation of roles exists between the voting commission's admin and the addresses and tokens distributor. The distributor knows to whom he sent each Ethereum address but does not know how that person votes. The voting committee knows how each Ethereum address voted but does not know the name of the person behind that address. Other people who know the Token's address can see which addresses voted (if they saved the holders list before the vote and compare it to the current list of holders) but they do not know who those addresses belong to or how they voted.

There is a tunable trade-off between privacy and transparency. For example, if the transactions are not visible (as in public anonymous platforms), the privacy will be better, and the transparency (and auditability) will be lessened as we will not be able to cross-check the total number of votes announced by the voting committee. The same will happen if the distributor does not store the correspondence between the Ethereum addresses and the voters, as one will not be able to check, if needed, if only the eligible voters (and all of them) received a token.

There are also some **limitations** of this approach:

- The Goerli network may not work properly or may be down during a voting session. One solution would be to use another network, such as Sepolia. This solution might delay the voting process as addresses and tokens need to be redistributed, but it does not require additional costs or expertise. Another solution is to use a plasma chain, a layer 2 solution that would be connected through a bridge to the Ethereum Mainnet. This solution increases centralization and requires extra development and ETH gas fees if some of the data are stored on the Mainnet.
- The voters need to have some IT skills to install a wallet and import the Ethereum address and the token.
- There is some organizational overhead regarding generating addresses, tokens, and distributing them. However, the approach is less complicated than others, as it does not require additional IoT devices [12] and offers verifiable security, transparency, and privacy, in contrast to DirectVote (https://www.surveyandballotsystems.com/directvote/), which offers restricted user-side transparency and does not seem to use BC.

## 6. Conclusions

In this paper, the main SC platforms have been discussed and a framework for BC-based voting is proposed. Section 5 shows how the solution addresses the research questions from the introduction. BC voting can be a viable alternative to in-person voting and mail-in voting during low-stakes elections. BC voting might be too risky for high-stakes polls at this time, as SC and BC still show some vulnerabilities, and even a rumor might considerably shake the public's trust in the system. Although it has its limitations, and future improvements are possible, it represents an improvement over traditional e-voting, where the user logs on to a website and votes.

During the production and maintenance phase, data could be collected. Analyzing such data can lead to system improvements, including developing voting fraud detection algorithms.

As another future development, zero-knowledge identity proof and homomorphic encryption could be added to the solution so that a voter could prove his eligibility to vote without revealing any personal information. Implementing such proofs could improve end-to-end verifiability and privacy, and mediate access to other university online services. As a limitation, zero-knowledge proofs are still slow compared to traditional encryption so they might be impractical to implement at scale.

To further improve trust, future research on formal verification of smart contracts is needed to establish industry-acceptable standards for checking that the SC code is free of errors and vulnerabilities.

A limitation of standard smart contracts is that they only can interact with data on their native blockchain. Smart contracts often require access to external data or the ability to trigger external actions (e.g., activate or deactivate a web page when the voting starts or ends). The term hybrid smart contract is sometimes used to describe a contract that can access external data using oracles. It might be possible to improve the functionality of smart contracts, including end-to-end verifiability, with further research into oracle systems. These systems provide a secure and reliable way to access external data and trigger external actions using on-chain code.

As shown in Sections 2 and 3, blockchain is a very dynamic and innovative research topic. It is dynamic to the point that it deters large- or even medium-scale implementations, as many fear that the technology stack they choose now might be deprecated even in a few months as other powerful solutions emerge into the market. Most BC approaches and solutions are vendor-driven and many of these vendors are startups. Although there has been an ISO technical committee for blockchain and distributed ledger technologies (ISO/TC 307) since 2016, there are still no published standards (December 2022). At least data exchange and identity formats should have been available to promote interoperability across technologies.

Although there are many papers discussing blockchain-based electronic voting systems, very few to none discuss the experience gained from using such systems in practice, so this would be a very interesting topic for future research. The lack of research analyzing actual implementations is also a problem for other BC use cases such as energy trading, logistics, or other legal agreements. More research based on running implementations would help in testing (by using empirical data) the potential of BC to make an impact in these fields and would help the development of policies beyond cryptocurrencies.

**Author Contributions:** All three authors contributed to the completion of the research. Conceptualization, V.D.; methodology, V.D.; software, V.D. and M.G.S.; validation, V.D.; formal analysis, V.D.; investigation, V.D.; resources, V.D., A.B. and M.G.S.; data curation, V.D.; writing—original draft preparation, V.D., A.B. and M.G.S.; writing—review and editing, A.B.; visualization, V.D., A.B. and M.G.S.; supervision; V.D.; project administration, V.D.; funding acquisition, V.D. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by a grant offered by the Bucharest University of Economic Studies, project ID 323/2022, project title "Electronic voting secured by blockchain technology—applicability in university elections" EN/"Votul electronic securizat prin tehnologia blockchain—aplicabilitate în alegerile din cadrul universităților" RO.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

## Appendix A. Example of a JSON File Containing an Ethereum Address

{"address":"0 × 9005adfa1e78b64be318c3b7ba86dd68ec40922a", "crypto": {"kdf": "pbkdf2", "kdfparams":{"c":262144, "dklen":32, "prf": "hmac-sha256", "salt": "45aee9fa6e8539334a02b040c8938ce19da2ca94320a29f3ce599c07a7453b4d"}, "cipher": "aes-128-ctr", "ciphertext": "536c5ad42c322cd7b01d71815767c5ae0a05ae28a3c134341078278-5dc9d2853", "cipherparams":"iv": "39df2ffc5f99511d344416ff3861bf0d"}, "mac": "2f77fcb9-e76fb08eedaa8ff57774a3fa35ede595c7d029c16776d540ce820000"}, "id": "205ec860-b376-4c64-8c7d-1c2499fa7f3d", "version":3}.

## References


1. Marella, V.; Upreti, B.; Merikivi, J.; Tuunainen, V.K. Understanding the Creation of Trust in Cryptocurrencies: The Case of Bitcoin. *Electron. Mark.* **2020**, *30*, 259–271. [CrossRef]
2. Atzei, N.; Bartoletti, M.; Cimoli, T. A Survey of Attacks on Ethereum Smart Contracts (SoK). In *International Conference on Principles of Security and Trust*; Springer: Uppsala, Sweden, 2017; pp. 164–186.
3. Lankton, N.; McKnight, D.H.; Thatcher, J.B. Incorporating Trust-in-Technology into Expectation Disconfirmation Theory. *J. Strateg. Inf. Syst.* **2014**, *23*, 128–145. [CrossRef]
4. Macrinici, D.; Cartofeanu, C.; Gao, S. Smart Contract Applications within Blockchain Technology: A Systematic Mapping Study. *Telemat. Inform.* **2018**, *35*, 2337–2354. [CrossRef]
5. Su, L.; Cao, Y.; Li, H.; Tan, J. Blockchain-Driven Optimal Strategies for Supply Chain Finance Based on a Tripartite Game Model. *J. Theor. Appl. Electron. Commer. Res.* **2022**, *17*, 1320–1335. [CrossRef]
6. Leng, J.; Zhou, M.; Zhao, J.L.; Huang, Y.; Bian, Y. Blockchain Security: A Survey of Techniques and Research Directions. *IEEE Trans. Serv. Comput.* **2022**, *15*, 2490–2510. [CrossRef]
7. Caldarelli, G.; Ellul, J. Trusted Academic Transcripts on the Blockchain: A Systematic Literature Review. *Appl. Sci.* **2021**, *11*, 1842. [CrossRef]
8. Jansen, M.; Hdhili, F.; Gouiaa, R.; Qasem, Z. Do Smart Contract Languages Need to Be Turing Complete? In *International Congress on Blockchain and Applications*; Springer: Cham, Switzerland; L´Aquila, Italy, 2020; pp. 19–26.
9. What-Is-Ethereum. Available online: https://www.simplilearn.com/tutorials/blockchain-tutorial/what-is-ethereum (accessed on 5 April 2022).
10. Blockchain Software Development Using the Ethereum-Network. Available online: https://www.devteam.space/blog/blockchain-software-development-using-the-ethereum-network/ (accessed on 5 April 2022).
11. McCorry, P.; Shahandashti, S.F.; Hao, F.; Haber, S.; Stornetta, W.S. A Smart Contract for Boardroom Voting with Maximum Voter Privacy. In *Financial Cryptography and Data Security*; Springer: Cham, Switzerland, 2017; Volume 10322, pp. 357–375. Available online: https://www.doi.org/10.1007/978-3-319-70972-7_20 (accessed on 27 December 2022).
12. Toma, C.; Popa, M.; Boja, C.; Ciurea, C.; Doinea, M. Secure and Anonymous Voting D-App with IoT Embedded Device Using Blockchain Technology. *Electronics* **2022**, *11*, 1895. [CrossRef]
13. Park, S.; Specter, M.; Narula, N.; Rivest, R.L. Going from Bad to Worse: From Internet Voting to Blockchain Voting. *J. Cybersecurity* **2021**, *7*, tyaa025. [CrossRef]
14. Rivest, R.L. On the Notion of 'Software Independence' in Voting Systems. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **2008**, *366*, 3759–3767. [CrossRef]
15. Bonica, A.; Grumbach, J.M.; Hill, C.; Jefferson, H. All-Mail Voting in Colorado Increases Turnout and Reduces Turnout Inequality. *Elect. Stud.* **2021**, *72*, 102363. [CrossRef]
16. Vogelsteller, F.; Buterin, V. ERC-20 Token Standard. Available online: https://eips.ethereum.org/EIPS/eip-20 (accessed on 26 December 2022).
17. Page, M.J.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan, S.E.; et al. The PRISMA 2020 Statement: An Updated Guideline for Reporting Systematic Reviews. *BMJ* **2021**, *10*, n71. [CrossRef]
18. Christidis, K.; Devetsikiotis, M. Blockchains and Smart Contracts for the Internet of Things. *IEEE Access* **2016**, *4*, 2292–2303. [CrossRef]
19. Da Xu, L.; Xu, E.L.; Li, L. Industry 4.0: State of the Art and Future Trends. *Int. J. Prod. Res.* **2018**, *56*, 2941–2962. [CrossRef]
20. Zheng, Z.; Xie, S.; Dai, H.N.; Chen, X.; Wang, H. Blockchain Challenges and Opportunities: A Survey. *Int. J. Web Grid Serv.* **2018**, *14*, 352. [CrossRef]
21. Khan, M.A.; Salah, K. IoT Security: Review, Blockchain Solutions, and Open Challenges. *Futur. Gener. Comput. Syst.* **2018**, *82*, 395–411. [CrossRef]
22. Saberi, S.; Kouhizadeh, M.; Sarkis, J.; Shen, L. Blockchain Technology and Its Relationships to Sustainable Supply Chain Management. *Int. J. Prod. Res.* **2019**, *57*, 2117–2135. [CrossRef]
23. Yli-Huumo, J.; Ko, D.; Choi, S.; Park, S.; Smolander, K. Where Is Current Research on Blockchain Technology?—A Systematic Review. *PLoS ONE* **2016**, *11*, e0163477. [CrossRef]
24. Tschorsch, F.; Scheuermann, B. Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 2084–2123. [CrossRef]
25. Andoni, M.; Robu, V.; Flynn, D.; Abram, S.; Geach, D.; Jenkins, D.; McCallum, P.; Peacock, A. Blockchain Technology in the Energy Sector: A Systematic Review of Challenges and Opportunities. *Renew. Sustain. Energy Rev.* **2019**, *100*, 143–174. [CrossRef]
26. Iansiti, M.; Lakhani, K.R. The Truth about Blockchain. *Harv. Bus. Rev.* **2017**. Available online: https://hbr.org/2017/01/the-truth-about-blockchain (accessed on 2 January 2023).
27. Mengelkamp, E.; Gärttner, J.; Rock, K.; Kessler, S.; Orsini, L.; Weinhardt, C. Designing Microgrid Energy Markets. *Appl. Energy* **2018**, *210*, 870–880. [CrossRef]
28. Kshetri, N.; Voas, J. Blockchain-Enabled E-Voting. *IEEE Softw.* **2018**, *35*, 95–99. [CrossRef]
29. Huang, J.; He, D.; Chen, Y.; Khan, M.K.; Luo, M. A Blockchain-Based Self-Tallying Voting Protocol With Maximum Voter Privacy. *IEEE Trans. Netw. Sci. Eng.* **2022**, *9*, 3808–3820. [CrossRef]

30. Jafar, U.; Ab Aziz, M.J.; Shukur, Z.; Hussain, H.A. A Systematic Literature Review and Meta-Analysis on Scalable Blockchain-Based Electronic Voting Systems. *Sensors* **2022**, *22*, 7585. [CrossRef] [PubMed]
31. Neziri, V.; Shabani, I.; Dervishi, R.; Rexha, B. Assuring Anonymity and Privacy in Electronic Voting with Distributed Technologies Based on Blockchain. *Appl. Sci.* **2022**, *12*, 5477. [CrossRef]
32. Pawlak, M.; Poniszewska-Marańda, A. Trends in Blockchain-Based Electronic Voting Systems. *Inf. Process. Manag.* **2021**, *58*, 102595. [CrossRef]
33. Curran, K. E-Voting on the Blockchain. *J. Br. Blockchain Assoc.* **2018**, *1*, 4451. [CrossRef]
34. Taş, R.; Tanrıöver, Ö.Ö. A Systematic Review of Challenges and Opportunities of Blockchain for E-Voting. *Symmetry* **2020**, *12*, 1328. [CrossRef]
35. Ayed, A. Ben A Conceptual Secure Blockchain-Based Electronic Voting System. *Int. J. Netw. Secur. Its Appl.* **2017**, *9*, 1–9.
36. Shukla, S.; Thasmiya, A.N.; Shashank, D.O.; Mamatha, H.R. Online Voting Application Using Ethereum Blockchain. In Proceedings of the 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, India, 19–22 September 2018; pp. 873–880.
37. Yi, H. Securing E-Voting Based on Blockchain in P2P Network. *EURASIP J. Wirel. Commun. Netw.* **2019**, *2019*, 137. [CrossRef]
38. Biswas, M.; Mahi, M.; Nayeen, J.; Hossen, R.; Acharjee, U.K.; Md, W. Buvots: A Blockchain Based Unmanipulated Voting Scheme. In Proceedings of the Proceedings of the 2nd International Conference on IoT, Social, Mobile, Analytics & Cloud in Computational Vision & Bio-Engineering (ISMAC-CVB 2020), Tiruchengodu, India, 29–30 October 2020.
39. Nasir, M.H.; Arshad, J.; Khan, M.M.; Fatima, M.; Salah, K.; Jayaraman, R. Scalable Blockchains—A Systematic Review. *Futur. Gener. Comput. Syst.* **2022**, *126*, 136–162. [CrossRef]
40. Szabo, N. Formalizing and Securing Relationships on Public Networks. *First Monday* **1997**, *2*, 9. [CrossRef]
41. Shrimali, B.; Patel, H.B. Blockchain State-of-the-Art: Architecture, Use Cases, Consensus, Challenges and Opportunities. *J. King Saud Univ. Comput. Inf. Sci.* **2022**, *34*, 6793–6807. [CrossRef]
42. Nguyen, C.T.; Hoang, D.T.; Nguyen, D.N.; Niyato, D.; Nguyen, H.T.; Dutkiewicz, E. Proof-of-Stake Consensus Mechanisms for Future Blockchain Networks: Fundamentals, Applications and Opportunities. *IEEE Access* **2019**, *7*, 85727–85745. [CrossRef]
43. Buterin, V. The Merge Will Reduce Worldwide Electricity Consumption by 0.2%. Available online: https://twitter.com/VitalikButerin/status/1570299062800510976 (accessed on 15 September 2022).
44. POS. Available online: https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/ (accessed on 2 January 2023).
45. Bhujel, S.; Rahulamathavan, Y. A Survey: Security, Transparency, and Scalability Issues of NFT’s and Its Marketplaces. *Sensors* **2022**, *22*, 8833. [CrossRef] [PubMed]
46. Hafid, A.; Hafid, A.S.; Samih, M. Scaling Blockchains: A Comprehensive Survey. *IEEE Access* **2020**, *8*, 125244–125262. [CrossRef]
47. Pierro, G.A.; Tonelli, R. Can Solana Be the Solution to the Blockchain Scalability Problem? In Proceedings of the 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), Honolulu, HI, USA, 15–18 March 2022; pp. 1219–1226.
48. Bodziony, N.; Jemioło, P.; Kluza, K.; Ogiela, M.R. Blockchain-Based Address Alias System. *J. Theor. Appl. Electron. Commer. Res.* **2021**, *16*, 1280–1296. [CrossRef]
49. Cui, S.; Zhao, G.; Gao, Y.; Tavu, T.; Huang, J. VRust: Automated Vulnerability Detection for Solana Smart Contracts. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security; Association for Computing Machinery: New York, NY, USA, 2022; pp. 639–652.
50. Hanke, T.; Movahedi, M.; Williams, D. DFINITY Technology Overview Series, Consensus System. *arXiv* **2018**, arXiv:1805.04548.
51. Voshmgir, S. *Token Economy: How the Web3 Reinvents the Internet*; BlockchainHub: Berlin, Germany, 2020.
52. Tu, S.-F.; Hsu, C.-S.; Wu, Y.-T. A Loyalty System Incorporated with Blockchain and Call Auction. *J. Theor. Appl. Electron. Commer. Res.* **2022**, *17*, 1107–1123. [CrossRef]
53. Duan, J.; Gu, L.; Zheng, S. ARCT: An Efficient Aggregating Ring Confidential Transaction Protocol in Blockchain. *IEEE Access* **2020**, *8*, 198118–198130. [CrossRef]
54. Yuen, T.H.; Sun, S.-F.; Liu, J.K.; Au, M.H.; Esgin, M.F.; Zhang, Q.; Gu, D. RingCT 3.0 for Blockchain Confidential Transaction: Shorter Size and Stronger Security. In *International Conference on Financial Cryptography and Data Security*; Springer: Cham, Switzerland, 2020; pp. 464–483.
55. Zhang, Z.; Li, W.; Liu, H.; Liu, J. A Refined Analysis of Zcash Anonymity. *IEEE Access* **2020**, *8*, 31845–31853. [CrossRef]
56. Kappos, G.; Yousaf, H.; Maller, M.; Meiklejohn, S. An Empirical Analysis of Anonymity in Zcash. In Proceedings of the 27th USENIX Security Symposium, Baltimore, MD, USA, 15–17 August 2018.
57. Li, T.; Wang, H.; He, D.; Yu, J. Blockchain-Based Privacy-Preserving and Rewarding Private Data Sharing for IoT. *IEEE Int. Things J.* **2022**, *9*, 15138–15149. [CrossRef]
58. Akcora, C.G.; Gel, Y.R.; Kantarcioglu, M. Blockchain Networks: Data Structures of Bitcoin, Monero, Zcash, Ethereum, Ripple, and Iota. *WIREs Data Min. Knowl. Discov.* **2022**, *12*, e1436. [CrossRef]
59. Rinberg, R.; Agarwal, N. Privacy When Everyone Is Watching: An SOK on Anonymity on the Blockchain, Cryptology ePrint Archive, Paper 2022/985. 2022. Available online: https://eprint.iacr.org/2022/985 (accessed on 2 January 2023).
60. Swathi, P.; Venkatesan, M. Scalability Improvement and Analysis of Permissioned-Blockchain. *ICT Express* **2021**, *7*, 283–289. [CrossRef]

61. Jiang, Y.; Lian, Z. High Performance and Scalable Byzantine Fault Tolerance. In Proceedings of the 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, China, 15–17 March 2019; pp. 1195–1202.
62. Barger, A.; Manevich, Y.; Meir, H.; Tock, Y. A Byzantine Fault-Tolerant Consensus Library for Hyperledger Fabric. In Proceedings of the 2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Sydney, Australia, 3–6 May 2021; pp. 1–9.
63. Chen, J.; Xue, J.; Wang, Y.; Huang, L.; Baker, T.; Zhou, Z. Privacy-Preserving and Traceable Federated Learning for Data Sharing in Industrial IoT Applications. *Expert Syst. Appl.* **2023**, *213*, 119036. [CrossRef]
64. Bing, W.; Hui-ling, L.; Li, P. Optimized DPoS Consensus Strategy: Credit-Weighted Comprehensive Election. *Ain Shams Eng. J.* **2023**, *14*, 101874. [CrossRef]
65. Zhang, B.; Kong, L.; Li, Q.; Min, X.; Liu, Y.; Che, Z. EB-BFT: An Elastic Batched BFT Consensus Protocol in Blockchain. *Futur. Gener. Comput. Syst.* **2023**, *139*, 267–279. [CrossRef]
66. Qiu, C.; Aujla, G.S.; Jiang, J.; Wen, W.; Zhang, P. Rendering Secure and Trustworthy Edge Intelligence in 5G-Enabled IIoT Using Proof of Learning Consensus Protocol. *IEEE Trans. Ind. Inform.* **2023**, *19*, 900–909. [CrossRef]
67. Mohsenzadeh, A.; Jalaly Bidgoly, A.; Farjami, Y. A Fair Consensus Model in Blockchain Based on Computational Reputation. *Expert Syst. Appl.* **2022**, *204*, 117578. [CrossRef]
68. Taghavi, M.; Bentahar, J.; Otrok, H.; Bakhtiyari, K. A Reinforcement Learning Model for the Reliability of Blockchain Oracles. *Expert Syst. Appl.* **2023**, *214*, 119160. [CrossRef]
69. Ruan, H.; Gao, H.; Qiu, H.; Gooi, H.B.; Liu, J. Distributed Operation Optimization of Active Distribution Network with P2P Electricity Trading in Blockchain Environment. *Appl. Energy* **2023**, *331*, 120405. [CrossRef]
70. Bao, Z.; Tang, C.; Lin, F.; Zheng, Z.; Yu, X. Rating-Protocol Optimization for Blockchain-Enabled Hybrid Energy Trading in Smart Grids. *Sci. China Inf. Sci.* **2023**, *66*, 159205. [CrossRef]
71. Zhou, F.; Liu, Y. Blockchain-Enabled Cross-Border E-Commerce Supply Chain Management: A Bibliometric Systematic Review. *Sustainability* **2022**, *14*, 15918. [CrossRef]
72. Zhang, S.; Wang, L.; Xiong, H. Chaintegrity: Blockchain-Enabled Large-Scale e-Voting System with Robustness and Universal Verifiability. *Int. J. Inf. Secur.* **2020**, *19*, 323–341. [CrossRef]