Peer-to-Peer Networking and Applications manuscript No.
(will be inserted by the editor)

Blockchain Smart Contracts: Applications, Challenges, and Future Trends

Shafaq Naheed Khan · Faiza Loukil · Chirine Ghedira-Guegan · Elhadj Benkhelifa · Anoud Bani-Hani

Received: date / Accepted: date

Abstract In recent years, the rapid development of blockchain technology and cryptocurrencies has influenced the financial industry by creating a new crypto-economy. Then, next-generation decentralized applications without involving a trusted third-party have emerged thanks to the appearance of smart contracts, which are computer protocols designed to facilitate, verify, and enforce automatically the negotiation and agreement among multiple untrustworthy parties. Despite the bright side of smart contracts, several concerns continue to undermine their adoption, such as security threats, vulnerabilities, and legal issues. In this paper, we present a comprehensive survey of blockchain-enabled smart contracts from both technical and usage points of view. To do so, we present a taxonomy of existing blockchain-enabled smart contract solutions, categorize the included research papers, and discuss the existing smart contract-based studies. Based on the findings from the survey, we identify a set of challenges and open issues that need to be addressed in future studies. Finally, we identify future trends.

Keywords Smart contract · Blockchain technology · Peer-to-peer network · Survey · Systematic Review

S-N. Khan
Zayed University, UAE
E-mail: shafaq.khan@zu.ac.ae

F. Loukil
University of Lyon, University Jean Moulin Lyon 3, CNRS, LIRIS, Lyon, FRANCE
E-mail: faiza.loukil@liris.cnrs.fr

C. Ghedira-Guegan
University of Lyon, University Jean Moulin Lyon 3, iaelyon school of Management, CNRS, LIRIS, Lyon, FRANCE; E-mail: chirine.ghedira-guegan@univ-lyon3.fr

E. Benkhelifa
Staffordshire University, Stoke on Trent, UK; E-mail: e.benkhelifa@staffs.ac.uk

A. Bani-Hani
Zayed University, UAE; E-mail: anoud.bani-hani@zu.ac.ae

1 Introduction

For more than a decade, the blockchain is established as a technology where a distributed database records all the transactions that have happened in a peer-to-peer network. It is regarded as a distributed computing paradigm that successfully overcomes the issue related to the trust of a centralized party. Thus, in a blockchain network, several nodes collaborate among them to secure and maintain a set of shared transaction records in a distributed way without relying on any trusted party. In 2008, Satoshi Nakamoto introduced Bitcoin [69] that was the first proposed cryptocurrency introducing the blockchain as a distributed infrastructural technology. It allowed users to transfer securely crypto-currencies, known as “bitcoins” without a centralized regulator. Besides, Ethereum [16], NXT [71], and Hyperledger Fabric [4] were also proposed as blockchain-based systems used for the cryptocurrency. Unlike Bitcoin, they can use smart contracts, which have emerged as a new promising use case to automate the execution of contracts in a distributed environment.

Smart contracts are executable codes that run on top of the blockchain to facilitate, execute, and enforce an agreement between untrustworthy parties without the involvement of a trusted third-party [16]. Smart contracts gave network automation and the ability to convert paper contracts into digital contracts. Compared to traditional contracts, smart contracts enabled users to codify their agreements and trust relations by providing automated transactions without the supervision of a central authority [89]. In order to prevent contract tampering, smart contracts are copied to each node of the blockchain network. By enabling the execution of the operations by computers and services provided by blockchain platforms, human error could be reduced to avoid disputes regarding such contracts.

Although smart contracts have made progress in recent years, they still face many challenges. For instance, one infamous malicious attack took place in 2016 when the Decentralized Autonomous Organization (DAO) smart contract was manipulated to steal around 2 Million Ether [1](50 Million USD on the time) because of its re-entrancy vulnerability [103]. In addition to the vulnerability problem, smart contracts face several challenges including privacy, legal, and performance issues.

To understand current topics on smart contracts, we conduct a comprehensive survey, with the aim of better identifying and mapping research areas that need further studies. The focus of this survey is studying smart contracts from the technical point of view (e.g., codifying, security, performance issues) and the usage point of view (e.g., smart contract applications in finance, healthcare, etc). The major contributions of this paper are summarized as follows:

1. We propose a taxonomy of studies based on blockchain-enabled smart contracts, which includes two main categories, namely smart contract improvement and smart contract usage.

[1] Ether (ETH): the cryptocurrency of Ethereum apps that is digital, global money.

2. We categorize 200 papers that we have extracted from different digital databases and discuss the existing smart contract-based studies.
3. Based on the findings from the survey, we identify a set of smart contract challenges and open issues that need to be addressed in future studies. Therefore, this survey provides a helpful reference to the researchers who want to target smart contract improvement or usage in their future studies.
4. Finally, we discuss future trends of smart contracts and explain how they provide better solutions to the open research challenges.

The remainder of this paper is structured as follows. Section 2 discusses background information about blockchain and smart contracts technologies. Section 3 discusses existing reviews studying smart contract-based approaches. Section 4 describes the adopted survey methodology and the solution taxonomy used to categorize existing smart contract-based solutions. In Sections 5-8, we present existing advances in modeling-driven smart contract improvement, optimization-driven smart contract improvement, resource-driven smart contract usage, and cross-organizational collaboration-driven smart contract usage. Section 9 discusses the study results by introducing challenges and future trends in the studied field. Finally, Section 10 concludes the paper.

2 Background

As aforementioned, blockchain technology has emerged as a distributed computing paradigm that successfully overcomes the problem related to the trust of a centralized party. Thus, in a blockchain network, several nodes collaborate among them to secure and maintain a set of shared transaction records in a distributed way without relying on any trusted party. Specific nodes in the network known as miners are responsible for adding new blocks to a distributed public ledger known as the blockchain.

The first system was Bitcoin [69], which allowed users to transfer securely the currency (bitcoins) without a centralized regulator. In the blockchain network, miners are responsible for collecting transactions, solving challenging computational puzzles (proof-of-work) in order to reach consensus, and adding the transactions as blocks to the blockchain. Since then, several blockchain-based development platforms have been proposed offering the ability to host/use smart contracts to execute automatically events and actions, namely NXT [71], Ethereum [16], Hyperledger Fabric [4], etc.

We detail below the smart contract operational process and then discuss some blockchain platforms that support the development of smart contracts.

2.1 Operational process of smart contracts

A smart contract is a common agreement between two or more parties. It stores information, processes inputs, and writes outputs thanks to its pre-defined functions [16]. For instance, the smart contract can define the constructor function that enables the smart contract creation. Hosting a new smart contract in the blockchain is enabled by invoking the constructor function through a transaction, whose sender becomes the smart contract owner. A self-destruct function is another example of the functions that can be defined in a smart contract. Usually, only the smart contract owner can destruct the contract by invoking this function.

A smart contract is likely to be a class that includes state variables, functions, function modifiers, events, and structures [16] which is intended to execute and control relevant events and actions according to the contract terms. Besides, it can even call other smart contracts. Each smart contract includes states and functions. The former are variables that hold some data or the owner's wallet address (i.e., the address in which the smart contract is deployed). We can distinguish between two state types, namely *constant states*, which can never be changed, and *writable states*, which save states in the blockchain. The latter are pieces of code that can read or modify states. We can distinguish between two function types, namely *read-only functions*, which do not require *gas* [2] to run and *write functions* that require *gas* because the state transitions must be encoded in a new block of the blockchain. Furthermore, paying currency is required to avoid infinitely smart contract runs.

As aforementioned, a smart contract is hosted in the blockchain by invoking its constructor function through a transaction submitted to the blockchain network, then the constructor function is executed, and the final code of the smart contract is stored on the blockchain. Once deployed, the creator of the smart contract got the returned parameters (e.g., contract address), then users can invoke any available smart contract's function by sending a transaction.

2.2 Platforms for Smart Contracts

Smart contracts can be developed and deployed in different blockchain platforms (e.g., NXT, Ethereum, and Hyperledger Fabric). Several platforms offer distinctive features for developing smart contracts including contract programming languages, contract code execution, and security levels. Some platforms support high-level programming languages to develop smart contracts.

- Bitcoin [69] is a public blockchain platform that can be used to process cryptocurrency transactions, but with a very limited computing capability. Bitcoin uses a stack-based bytecode scripting language. The ability to create a smart contract with rich logic using the Bitcoin scripting language is very limited. Major changes would need to be made to both the mining functions and the mining incentivization schemes to enable smart contracts proper on Bitcoin's blockchain [52].
- NXT [71] is an open-source blockchain platform that relies entirely on a proof-of-stake consensus protocol. It includes a selection of smart contracts

[2] gas: a unit that measures the amount of computational effort that it will take to execute certain operations.

that are currently living. However, it is not Turing-complete, meaning only the existing templates can be used and no personalized smart contract can be deployed.

- Ethereum [16] is the first blockchain platform for developing smart contracts. It supports advanced and customized smart contracts with the help of a Turing-complete virtual machine, called the Ethereum virtual machine (EVM). EVM is the runtime environment for smart contracts, and every node in the Ethereum network runs an EVM implementation and executes the same instructions. Solidity, as a high-level programming language, is used to write smart contracts, and the contract code is compiled down to EVM bytecode and deployed on the blockchain for execution. Ethereum is currently the most popular development platform for smart contracts and can be used to design various kinds of decentralized applications (DApps) in several domains.
- Rather than the public blockchain, such as Bitcoin and Ethereum that any party can participate in the network, Hyperledger Fabric [4] is permissioned with only a collection of business-related organizations can join in through a membership service provider, and its network is built up from the peers whose are owned and contributed by those organizations. Hyperledger Fabric is an open-source enterprise-grade distributed ledger technology platform, proposed by IBM and supports smart contracts. It offers modularity and versatility for a broad set of industry use cases. The modular architecture for Hyperledger Fabric accommodates the diversity of enterprise use cases through plug and play components.

Ethereum and Hyperledger Fabric smart contracts differ in multiple aspects. While Solidity is the well-known programming language used to write Ethereum smart contracts, Hyperledger Fabric supports multi-language smart contracts, such as Go, Java, and Javascript [4]. For contract code execution, the contract code in Ethereum is included in a transaction, which is propagated in the peer-to-peer network, and any miner that receives this transaction can execute it in its local virtual machine [16]. In Hyperledger Fabric, when a transaction is created by the application, the transaction is only executed and signed by specified peers (endorsing peers). After receiving the application's transaction proposal, each of these endorsing peers independently executes it by invoking the chain-code to which the transaction refers [4]. For security, chaincode runs within a container environment (e.g., Docker) for isolation.

These blockchain-based development platforms are used in the existing studies that we detail in the following sections.

3 Related literature reviews/surveys

We provide a brief overview of the existing reviews that have studied blockchain-enabled smart contracts.

While several literature reviews/surveys are published in order to study the blockchain-enabled smart contracts, there are still some ongoing challenges

Table 1: Existing Smart Contract Reviews/Surveys: A Comparative Summary

Survey	Solution taxonomy?	Blockchain platforms?	Application domains?	Coverage of tools?	Research gap identification?	Scope of literature review
Our Survey	✓	✓	✓	✓	✓	Until 09/2020
[8]	✓	✓		✓		Until 2017
[25]		✓		✓		Until 09/2018
[5]		✓		✓		Until 10/2018
[56]	✓	✓		✓		2015-2018
[33]		✓		✓		Until 05/2019
[68]		✓		✓		Until 2019
[120]				✓		Until 2019
[37]	✓			✓		2015-01/2020
[77]	✓	✓		✓	✓	Until 2019
[66]		✓	✓			Until 2018
[81]		✓	✓			Until 04/2019
[12]	✓	✓	✓			2013-2016
[63]			✓			Until 2018
[41]		✓	✓			2015-2019
[60]		✓				Until 2018
[21]		✓	✓			Until 05/2016
[1]		✓	✓			Until 05/2017
[95]			✓			2013-2018
[99]		✓	✓		✓	Until 2018
[117]		✓	✓		✓	Until 2019

that have not been addressed. Table 1 presents a comparative summary of the existing blockchain-enabled smart contract reviews/surveys according to six criteria, namely proposing a taxonomy, considering several blockchain platforms, considering application domains, covering smart contract improvement tools, identifying research gaps, and scope of literature review. We observe that there is a lack of taxonomy focusing on smart contract improvement (i.e., addressing smart contract security, privacy, and performance issues) and smart contract usage (i.e., addressing domain-specific issues).

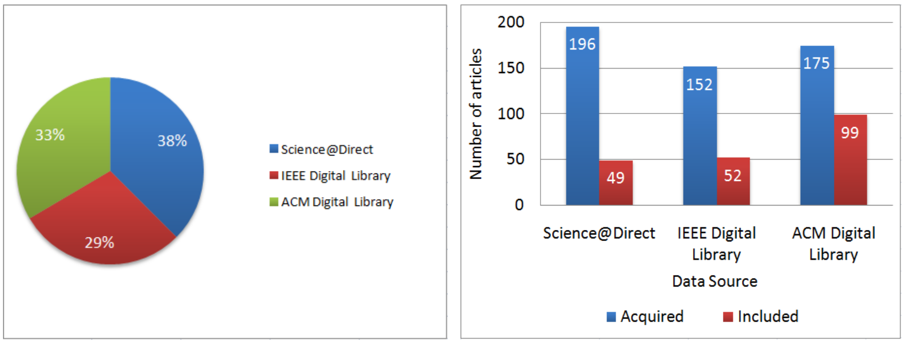
To sum up, it can be said that the existing surveys concerning blockchain-enabled smart contracts focus on classifying the papers based on smart contract issues. Our work extends the existing surveys by studying the smart contract application domains, analyzing the smart contract challenges, and introducing some research gaps that need to be addressed in future studies.

4 Research Methodology and Solution Taxonomy

We describe below the adopted research methodology, such as the search strategy, filtering process, and inclusion and exclusion criteria. Besides, we present the solution taxonomy used to categorize the final set of included papers.

4.1 Systematic Literature Review

We used three existing databases, namely ScienceDirect, IEEEXplore, and ACM Digital Library to search for relevant works using the "smart contract" string keyword. In the first phase, we found 523 publications as shown in both Figure 1a, which depicts the percentage of the acquired research paper per digital database as well as Figure 1b, which depicts the total number of preliminary studies acquired from each digital database.



(a) Acquired Articles Per Source

(b) Acquired/Included Articles Per Source

Fig. 1: Publication trend

Table 2: Inclusion and exclusion criteria for relevant works

Inclusion criteria	**Exclusion criteria**
Be published online before September 2020	White papers, editorial comments, and book reviews
Studies are in the field of smart contracts	Studies that present surveys and review papers
Studies that are available in online archives	Studies that are not published in English

To choose the relevant papers to be analyzed in our review, we filtered the primary studies retrieved from the databases. To do so, we defined a set of inclusion and exclusion criteria, which are summarised in Table 2. Based on the outcomes of the first phase, we applied the set of inclusion and exclusion criteria to exclude the publications considered outside the scope of this review. Thus, we only included studies that satisfy all the inclusion criteria. We excluded duplicate publications, surveys, and literature reviews by filtering studies based on the title, the abstract, and the list of keywords.

As a result of the filtering process, we excluded 323 publications and included 200 relevant publications for this systematic review. Figure 1b depicts also the number of the relevant studies included in this research from each digital database.

4.2 Publication trends and Categorization

To examine the trend of the smart contract field in terms of the publication date, Figure 2 depicts the number of included studies published each year from 2015 to September 2020. We observe that the total number of published papers in the studied field increases in the past few years, indicating the importance of the topic. Thus, the smart contract field is rapidly growing in recent years.

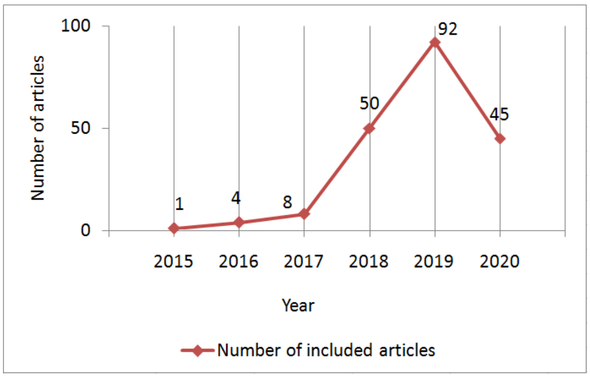


Fig. 2: Included Articles Per Year

As a result of an in-depth analysis of the included studies in this review, a comprehensive taxonomy is constructed to provide an additional support for designers to understand the various dimensions that they have to consider when designing a smart contract. The major motivations of this survey are to identify (i) the main publications about smart contracts, (ii) the current state of research in this field, and (iii) possible gaps in the literature that could become research problems to be solved by the scientific community. This survey is not useful just to define the conceptual background of blockchain-enabled smart contracts, but also to identify research issues to be explored at new studies. Indeed, we categorize existing smart contract research into two major categories, namely smart contract improvement and smart contract usage. The former includes studies aiming at addressing the smart contract challenges, such as functionality verification, performance, vulnerabilities, and lack of trustworthy data feeding. The latter includes studies aiming at addressing domain-specific challenges using smart contracts. Figure 3 depicts the proposed taxonomy of blockchain-enabled smart contracts, including modeling-driven smart contract improvement (see Section 5), optimization-driven smart contract improvement (see Section 6), resource-driven smart contract usage (see Section 7), and cross-organizational collaboration-driven smart contract usage (see Section 8).

5 Modeling-driven smart contract improvement

Smart contracts have suffered from multiple security vulnerabilities in the past few years [8], which have resulted in both theft and gigantic financial losses. Such vulnerabilities could have been avoided with the help of formal analysis and verification of such smart contracts before deploying them on the blockchain. Since existing programming languages, such as Solidity are not built for formal verification, several researchers have proposed alternative approaches in order to improve the smart contract functionality verification.

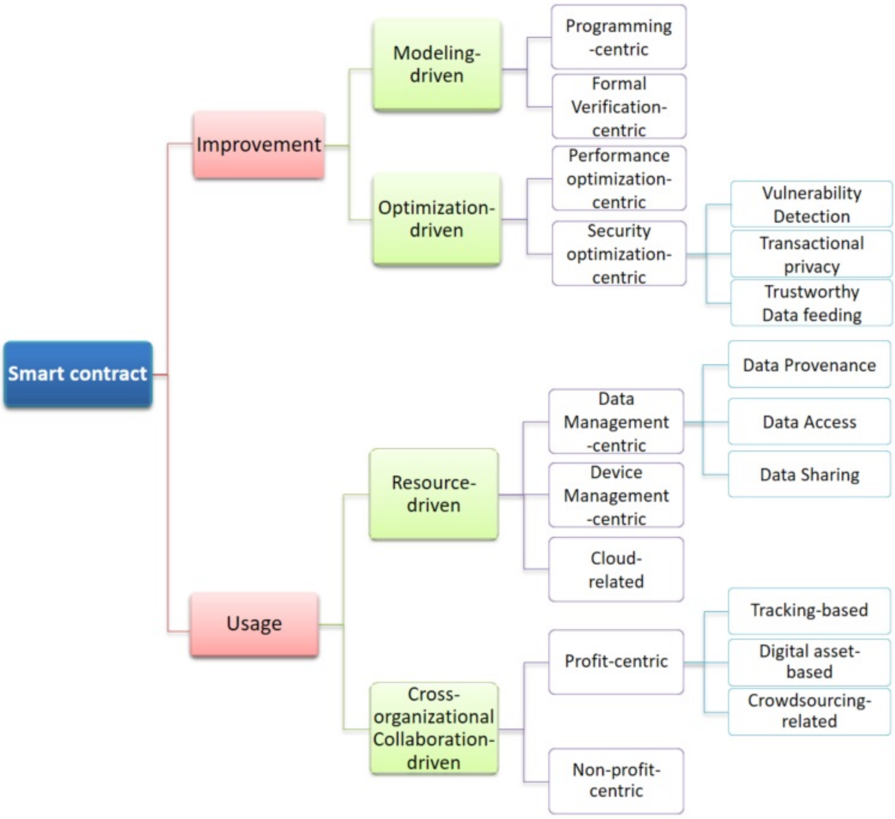


Fig. 3: Taxonomy of blockchain-enabled smart contract based studies.

In this category, we discussed modeling-driven smart contract improvement solutions, which can be categorized into programming-centric solutions (see Table 3) and formal verification-centric solutions (see Table 4).

5.1 Programming-centric solutions

The essence of a smart contract is the computer code that can be executed automatically on the computer, so programming smart contracts correctly is an important research direction. Several researchers argued that developing new contract languages is an effective way to write a correct smart contract. Table 3 presents some newly proposed programming languages such as SmaCoNat [78], Flint [83], and Scilla [85]. For instance, Regnath and Steinhorst [78] proposed a human-readable, security, and executable programming language, called SmaCoNat. The authors converted programming language grammar into natural language sentences in order to improve program readability.

Table 3: Some examples of programming-centric solutions

Paper	Contribution	Description
[78]	SmaCoNat	SmaCoNat is a domain-specific language that is tailored for a subset of the transaction logic found in smart contracts.
[83]	Flint	Flint is a type-safe, capabilities-secure, contractoriented programming language specifically designed for writing robust smart contracts.
[85]	Scilla	Scilla is a novel intermediate-level functional smart contract programming language, suitable to serve as a compilation target and also as an independent programming framework. Scilla aims at achieving both sufficient expressivity and tractability, while enabling formal contract verification.

New contract languages promised to address the existing domain-specific language vulnerabilities. However, since they have not been put into practice, they could have their vulnerabilities. Thus, designing and implementing secure smart contracts still require adaptive software engineering technologies and expertise from multiple research domains, such as networking, programming languages, formal methods, and cryptography.

5.2 Formal verification-centric solutions

Typically, formal testing is applied to ensure that a software behaves and performs as expected in its specifications and requirements based on all possible inputs' conditions. For smart contracts, Truffle [93], is an example of a development framework for Ethereum that enables writing formal test cases based on certain mathematical logic and rules for smart contracts written in JavaScript or Solidity languages. These test cases can be written in JavaScript and can be executed on a test network to check several properties of smart contracts. As aforementioned, formal testing can only make sure that a smart contract did what it is supposed to do based on its specification, however, it cannot help the smart contract developers to find bugs or vulnerabilities. Therefore, automated formal verification is a promising approach to detect bugs and other errors to guarantee the functional correctness of smart contracts. According to [2], formal verification can provide the highest level of confidence in the correct behavior of smart contracts. At present, the use of formal methods to verify smart contracts has been widely adopted by several researchers, and significant results have been achieved in practice. Table 4 presents some formal verification-centric solutions. For instance, Amani et al. [2] extended an existing EVM formalization in Isabelle/HOL by a sound program logic at the level of bytecode. The principle of the method is to organize the bytecode sequences into linear code blocks and create a logic program, where each block is processed as a set of instructions. Each part of the verification is validated in a single trusted logical framework from the perspective of bytecode.

Table 4: Some examples of formal verification-centric solutions

Publication	Contribution	Description
[2]	Bytecode verifying method	It aims at verifying smart contracts at the level of EVM bytecode using the Isabelle/HOL. This formal method is generic to all Ethereum smart contracts.
[10]	Model checking method	It is introduced based on formal methods to model smart contracts and verify their properties. Formal methods combined with smart contracts aim at reducing the potential errors and costs during the development process of smart contracts.
[73]	Tool chain	Tool chain aims at translating chain code modeled in Solidity via its operational semantics into a formal representation that can be formally analyzed for correct implementation via model checking.
[109]	FEther	FEther is an extensible hybrid verification proof engine for Ethereum smart contract verification. Based on Lolisa, which is a large subset of solidity mechanized in Coq, FEther guarantees the consistency between smart contracts and its formal model.

Currently, formal verification tools are still in the experimental stage and have not been widely used. Therefore, the smart contract formalization research direction deserves a lot of attention, thus it provides the highest level of confidence about the correct behavior of smart contracts. Real progress in this research field can improve trust in the smart contract, especially when used to develop critical systems, such as financial, healthcare, and banking systems.

6 Optimization-driven smart contract improvement

Smart contracts have emerged as a new promising solution for developing fully decentralized applications without involving a trusted third-party. Despite the bright side of smart contracts, several concerns continue to undermine their adoption, namely performance issues, security threats, and privacy issues. Indeed, new smart contract applications are more demanding in terms of contract execution time, execution cost, security, and privacy fields. In this category, we discuss optimization-driven smart contract improvement solutions, which can be categorized into performance optimization-centric solutions (see Table 5) and security optimization-centric solutions (see Table 6).

6.1 Performance optimization-centric solutions

Smart contract performance refers to the ability of smart contract systems to deliver in a reasonable response time and sustain performance when the number of contracts is increasing [1]. Table 5 presents some examples of performance optimization-centric solutions. Some performance issues in blockchain systems, not limited to, are throughput bottleneck, limited scalability, transactions latency. To overcome performance issues in smart contract systems,

Table 5: Some examples of performance optimization-centric solutions

Publication	**Contribution**	**Description**
[26]	Parallel execution method	It is a novel way to permit miners to execute smart contracts in parallel, based on techniques adapted from software transactional memory. This method performed well on smart contract benchmarks, greatly speeding up contract execution efficiency.
[34]	Parallel execution scheme	It can run multiple smart contracts in parallel to improve the throughput of the system.
[18]	GasReducer tool	GasReducer is a tool to automatically detect multiple anti-patterns from the bytecode of smart contracts and replace them with efficient code through bytecode-to-bytecode optimization in order to save gas cost.

some researchers have proposed solutions to execute smart contracts in parallel instead of sequentially [26][34]. For instance, Gao et al. [34] have proposed a parallel execution scheme that relies on two key techniques, namely a fair contract partition algorithm leveraging integer linear programming to partition a set of smart contracts into multiple subsets, and a random assignment protocol assigning subsets randomly to a subgroup of users. Other studies have been proposed for smart contract optimization by saving gas. In fact, if the smart contract execution exceeds an amount of gas (known as gas limit), an out-of-gas exception is raised, interrupting the current execution. For instance, GasReducer [18] is a tool for automatically detecting EVM operation sequences that can be replaced with other operations that have the same semantics but need less gas, and then replacing them with efficient code.

6.2 Security optimization-centric solutions

Security of a smart contract refers to its robustness against attacks from malicious users that exploit generally the contract security vulnerabilities to gain profit or the lack of trustworthy data feeding to inject malicious data. Table 6 presents some examples of vulnerability detection tools, transactional privacy models, and trustworthy data feeding solutions.

6.2.1 Vulnerability Detection

Discovering potential vulnerabilities in the execution of contracts is important to improve the security and credibility of contracts. Indeed, several studies systematically summarized the contract vulnerabilities and analyzed the security risks [8][77][81]. For instance, Atzei et al. [8] have provided a taxonomy of smart contract vulnerabilities of three levels, namely Solidity, EVM, and Blockchain. In recent years, the most notorious attack is the Decentralized Autonomous Organization (DAO) attack that exploited a re-entrancy vulnerability to steal

Table 6: Some examples of security optimization-centric solutions

Publication	Contribution	Description
		Vulnerability Detection
[59]	Oyente	Oyente is a symbolic execution tool that aims at finding potential security bugs. It extracted the control map from the EVM Bytecode of the contract and found potential vulnerabilities in the contract by executing a control map.
[15]	SmartInspect	SmartInspect is a solidity smart contract inspector that aims at analyzing contract states using decompilation techniques driven by the contract structure definition. It also allows contract developers to better visualize and understand the contract stored state without needing to redeploy, nor develop any ad-hoc code.
[47]	ContractFuzzer	ContractFuzzer is a novel fuzzer to test Ethereum smart contracts for security vulnerabilities. ContractFuzzer generates fuzzing inputs based on the ABI specifications of smart contracts, defines test oracles to detect security vulnerabilities, instruments the EVM to log smart contract run-time behaviors, and analyzes these logs to report security vulnerabilities.
[54]	ReGuard	ReGuard is a fuzzing-based analyzer to automatically detect re-entrancy bugs in Ethereum smart contracts. Specifically, ReGuard performs fuzz testing on smart contracts by iteratively generating random but diverse transactions.
[48]	EthRacer	EthRacer is an automatic analysis tool that runs directly on Ethereum bytecode and requires no hints from users in order to detect event-ordering bugs in blockchain smart contracts.
		Transactional privacy
[49]	Hawk	Hawk is a blockchain model of cryptography and privacy-preserving smart contracts. It does not make financial transactions available publicly on the blockchain to maintain transactional privacy.
[101]	Verifying contract protocol	It aims at deploying an encrypted smart contract on the blockchain. Only participants having a decryption key can access the contract's content.
		Trustworthy data feeding
[113]	Town Crier	Town Crier acts as a bridge between smart contracts and existing web sites, which are already commonly trusted for non-blockchain applications.
[57]	Data carrier architecture	Data carrier architecture is cost-effective and elastic for blockchain-enabled IoT environment that enables smart contracts to fetch off-chain data. The evaluation results show that the proposal is more efficient and elastic compared with Oraclize Oracle data carrier service.

around 2 Million Ether from a smart contract [103]. Another attack has happened to the SmartBillions, which presented a fully decentralized and transparent lottery system when an attacker successfully manipulated the block hash of the smart contract's lottery function twice, and forced the result in his favor to get 400 Ether [62]. To solve the smart contract vulnerabilities, several vulnerability detection solutions have been proposed. Some studies have given solutions to common vulnerabilities, such as Oyente [59], SmartInspect [15], and ContractFuzzer [47]. Some other work focused on specific vulnerabilities, such as ReGuard [54] to detect re-entrancy bugs and EthRacer [48] to detect event-ordering bugs.

6.2.2 Transactional privacy

The privacy issue represents a real challenge for smart contracts to keep critical functions secret, apply cryptography, and avoid disclosing data on the blockchain to the public. The lack of transactional privacy could limit the adoption of smart contracts. To address this issue, Kosba et al. [49] have proposed Hawk, a decentralized smart contract system. Hawk is a tool allowing smart contract developers to build privacy-preserving contracts without the need for implementing any cryptography. Its compiler automatically generated an efficient cryptographic protocol where contractual parties interact with the blockchain, using cryptographic primitives such as zero-knowledge proofs.

6.2.3 Trustworthy data feeding

The smart contract execution requires some external data about real-world states and events from outside the blockchain. Therefore, trustworthy data feeding mechanisms (known as Oracles) are required to build a bridge between blockchain and the external world (e.g., Web API). For instance, Zhang et al. [113] have proposed Town Crier, which acted as a link between existing commonly trusted non-blockchain based websites and smart contracts to provide authenticated data to smart contracts while preserving confidentiality with encrypted parameters. However, in case of malicious code or bad data fed to a smart contract, the latter processes the input as is, producing an incorrect and unpredictable outcome. Thus, oracles retain an enormous amount of power over smart contracts in how they are executed because the data they provide determines how the smart contracts execute.

To sum up, research on improving smart contract security and performance has emerged in recent years. While running smart contracts in parallel can speed up contract execution, it faces a challenge in how to execute contracts that depend on each other at the same time. Moreover, optimizing smart contract codes can effectively reduce potential vulnerabilities in contracts and ensure efficient and secure execution of contracts. However, the existing studies are still immature, and unknown vulnerabilities or bugs cannot be detected to be replaced. Thus, the optimization of smart contracts needs further research.

After discussing the smart contract from the technical point of view, we present in the following two sections the existing solutions focusing on smart contract usage in several domains.

7 Resource-driven smart contract usage

As we know, smart contracts are executable code hosted in the blockchain that store information, process inputs, and write outputs thanks to their pre-defined functions. They are used to improve data handling transparency, decentralize resource-constrained device management, and enable changes of the agreement terms at runtime while running on top of a decentralized and transparent network. In this category, we discuss resource-driven smart contract usage solutions, which can be categorized into data management-centric solutions (see Table 7), device management-centric solutions (see Table 8), and cloud-related solutions (see Table 9).

7.1 Data management-centric solutions

In the past, raw data are transferred to a cloud server to be stored and analyzed. However, this centralized solution has caused serious concerns regarding several aspects, such as the necessity to trust the cloud infrastructure security, control loss once data are externalized, and lack of data handling transparency. Consequently, blockchain-based data management emerged as a platform to facilitate transparent data transactions between untrustworthy involved parties on the network. Indeed, peer-to-peer-network-based data management is a more fair system as compared to a system where all transactions are handled by a central server. Table 7 presents some examples of data management-centric solutions concerning data provenance, data access, and data sharing.

7.1.1 Data Provenance

Data provenance refers to a historical record of the data and its origins showing which and how data item is stored, accessed, and processed by whom and for what purpose. Ensuring data provenance can increase data transparency and enforce data integrity. In this regard, a blockchain can offer an immutable storage of records and smart contracts can be used as a responsible for verifying the data origins before storing them. Similar ideas are applied in [6][44], where a blockchain is used as a decentralized and immutable storage for enabling data provenance. For instance, Javaid et al. [44] have proposed a blockchain-based data provenance and integrity for secure IoT environments framework, called BlockPro. Ethereum and two smart contracts were used to implement it. The first smart contract established data provenance by interacting with the IoT devices and making sure they are legit and the data being uploaded is coming from a known and trusted origin. The second smart contract can only be called by the first one to storing data on and retrieving data from the blockchain.

Table 7: Some examples of data management-centric solutions

Publication	Contribution	Description
		Data Provenance
[6]	FabRec	It is a decentralized approach to handle manufacturing information generated by various organizations using the blockchain. It decentralizes critical information about the manufacturer and makes it available on a peer-to-peer network composed of fiduciary nodes to ensure transparency and data provenance.
[44]	BlockPro	BlockPro is a solution based on Physical unclonable functions (PUFs) and the blockchain for a safe and secure IoT environment to ensure data provenance and enforce data integrity by providing an immutable storage platform.
		Data Access
[74]	FairAccess	FairAccess is a decentralized pseudonymous and privacy-preserving authorization management framework. It relies on smart contracts to express access control policies and blockchain to manage access control enforcement.
[36]	Multi-authority scheme	Multi-authority attribute-based access control (ABAC) scheme uses smart contract to issue a secret key to the data user to access the requested object.
[61]	Access control system	It aims at codifying attribute-based access control policies as smart contracts and deploying them on a blockchain, hence transforming the policy evaluation process into a completely distributed smart contract execution.
[114]	Access control framework	It is based on multiple access control contracts, one judge contract, and one register contract in order to achieve distributed and trustworthy access control for IoT systems.
[88]	AAA scheme	It is a blockchain-empowered Authentication/Authorization/Auditing (AAA) scheme to protect the data in the large-scale HetNet where the access control permission of data is stored on the blockchain.
[112]	AC scheme in IIoT	It is a blockchain-enhanced security access control scheme that supports traceability and revocability has been proposed in IIoT for smart factories.
		Data Sharing
[22]	Ancile	Ancile is a blockchain-based framework for secure, interoperable, and efficient access to medical records by patients, providers, and third parties while preserving the privacy of patients.
[72]	MediBchain	It is a patient-centric healthcare data management system using blockchain technology as storage which helps to attain privacy. Cryptographic functions are used to encrypt patient's data and to ensure pseudonymity.
[70]	Document sharing framework	Document sharing framework is a blockchain-based solution for document sharing and version control to facilitate multi-user collaboration and track changes. Smart contracts are used to govern and regulate the document version control functions among the creators of the document and its validators.
[30]	BABSC	BABSC is a blockchain-based attribute-based signcryption scheme to provide secure data sharing in the cloud environment. It also provides secure data confidentiality and unforgeability.
[111]	Research support platform	It is a blockchain-based platform for data sharing against COVID-19. Smart contracts and pseudonym mechanism are used to preserve the privacy of patients.

7.1.2 Data Access

Data access is ensured according to rights given to involved parties in a network to perform some operations on data. These rights are expressed using access control policies, which consist of a set of conditions that are evaluated against the current context to make the access decision each time a request is received [40]. Recently, for obtaining decentralized self-evaluating policies, access control policies have been codified as executable code and have been managed through a peer-to-peer network while eliminating a central entity. For this purpose, smart contracts can be used to express access control policies to transform the policy evaluation process into a distributed smart contract execution. In this context, several studies [74][36][61][114][88][112]have been proposed. For instance, Maesa et al. [61] proposed to exploit a blockchain to store access control policies and manage attributes, as well as to execute the access decision process. The access control policy is represented through a smart contract that evaluated the stored conditions to make the access decision.

7.1.3 Data Sharing

Data sharing refers to make data available to other parties by the data owner. However, two types of challenges faced data sharing schemes, namely (i) achieving good data sharing while losing the control over the shared data or (ii) remaining poor at sharing in order to keep strong control over the data. To address these challenges, blockchain technology is used because it offers immutable storage of records that improve data handling transparency and can host executable codes (i.e., smart contracts) that authenticate users, verify authorizations, and thereby ensure an efficient and secure data sharing in a peer-to-peer network. Several studies using blockchain-enabled smart contracts have been proposed for data sharing in healthcare [22][72][111], cloud environment [70], and for digital document version control [30]. In the healthcare context, medical devices and health care applications have been increasingly adopted by patients. However, wireless body sensors collect health records that are sensitive to individuals. Existing electronic health record management systems struggle with balancing data privacy and data access. Blockchain technology is an emerging technology that enables data sharing in a decentralized and transactional fashion. For instance, Dagher et al. [22] have proposed a blockchain-based framework, called Ancile for secure and efficient access to medical records by patients, providers, and third-parties while preserving the patients' privacy. Ancile employed smart contracts, data obfuscation techniques, and cryptographic techniques in order to improve privacy and security in the healthcare domain. Recently, Yu et al. [111] have proposed a blockchain-based medical research support platform, which employed the characteristics of the alliance chain on which hospitals and medical research institutions are treated as nodes. Among them, users such as patients, doctors, and researchers needed to register and authenticate on the alliance chain. Smart contracts are used to upload the pseudonymous addresses of CEMRs to the alliance chain.

Table 8: Some examples of device management-centric solutions

Publication	**Contribution**	**Description**
[29]	AlkylVM	AlkylVM is a split-virtual machine that allows for resource-constrained IoT devices to interact with blockchain systems.
[45]	IoT-Blockchain model	IoT-Blockchain model is an IoT device and server communication framework on Ethereum using a customized smart contract which enables a better defense mechanism against DDoS and rogue device attacks.
[58]	PrivBlockchain	PrivBlockchain is an end-to-end privacy-preserving framework for the IoT data using blockchain technology. The proposed smart contracts are used to improve the data ownership, transparency, and auditability for users.
[90]	LMS	Leave Management System (LMS) is a secure reliable leave management system through blockchain smart contract handled via mobile or IoT devices.
[96]	PoRX	Proof-of-Reputation-X (PoRX) is a reputation incentive scheme for blockchain consensus of Industrial Internet of Things.
[104]	SmartEdge	SmartEdge is an Ethereum-based smart contract for edge computing. It is a low-cost, low-overhead tool for compute-resource management.
[116]	Software update protocol	It is a blockchain based privacy-preserving protocol, which delivers secure and reliable updates for the IoT devices with an incentive mechanism while protects the privacy of involved users.

7.2 Device management-centric solutions

One of the technical challenges of having billions of devices deployed worldwide is the ability to manage and synchronize them. Using the current model of the server-client system may have some limitations for device management thus, several researchers are studying the benefits of the blockchain use in this field. Specifically, smart contracts are chosen to guarantee authentication, synchronization, and data integrity while running on top of a decentralized and transparent network. Table 8 presents some newly proposed device management-centric solutions [29][45][58][90][96][104][116]. For instance, Ellul and Pace [29] have proposed a split-virtual machine architecture to enable the integration of resource-constrained devices with blockchain systems, called AlkylVM. Each blockchain-connected device would run an instance of AlkylVM, which allows communication between blockchain and IoT devices using the Aryl blockchain node. The latter is responsible for monitoring smart contract transactions and events that would require interaction with IoT devices.

7.3 Cloud-related solutions

In cloud computing, both service requester and service provider agree on a set of requirements, obligations, and rights that is valid for the whole contract life-cycle. Recently, blockchain-enabled smart contracts have been used

Table 9: Some examples of cloud-related solutions

Publication	Contribution	Description
[42]	Automatic indemnification mechanism	It is based on smart contracts for refunding cloud storage service clients when the service provider violates the service level agreement by raising objections to a smart contract.
[84]	Smart Contract Negotiation	It is an autonomous negotiation of smart contracts in cloud computing, which analyses the cost and the necessary changes for reaching an agreement. It is based on a formal language that specifies interactions between offers and requests.
[98]	QoS-Aware Service Composition	It is a smart-contract based algorithm for constructing cloud service-based systems through the composition of existing services.
[118]	Cloud SLA Enforcement	It is a witness model to credibly enforce the cloud service level agreement (SLA) using the witness role based on blockchain and smart contracts to solve the trust issues about who can detect the service violation and how the violation is confirmed.

to enable changes in the agreement terms at runtime through the definition of conditions and actions. Table 9 presents some proposed cloud-related solutions [42][84][98][118]. For instance, Zhou et al. [118] have proposed a witness model for enforcing cloud Service Level Agreement (SLA) using smart contracts. The game theory is leveraged to analyze that the witness has to offer honest monitoring service in order to maximize its revenue. The service provider needs to prepay fees to the smart contract for hiring witnesses. The service customer then decides whether to accept the SLA. If yes, it also needs to prepay fees including the service fee and its part of the hiring fee for witnesses. However, a small bug or attack on smart contracts can result in significant issues like privacy leakage or system logic modifications. Some of critical security vulnerabilities can include timestamp dependence, mishandled exceptions, re-entrancy attacks on smart contracts in cloud-related solutions.

Although smart contracts fulfill many conditions related to data/device management, they have some drawbacks, based on basic design principles of blockchain technology. First, the data stored on smart contracts are publicly readable through public transactions with no read access restrictions. Thus, it is required to avoid storing private or device keys on smart contracts to the public availability of the information. For solving transparency problems related to blockchain, future research might investigate deploying complex cryptographic solutions for securing data stored on smart contracts without boosting cost. Second, the cost of storing data on the blockchain is very high. Therefore, creating hybrid solutions is required to benefit from the traceability of data transactions that are offered by blockchain networks and the efficient and private access and storage of data provided by external data repositories.

After discussing the resource-driven smart contract usage solutions, we present in the following section the existing solutions focusing on cross-organizational collaboration-driven smart contract usage.

8 Cross-organizational collaboration-driven smart contract usage

Smart contracts help to record an agreement between several untrustworthy parties in the form of code that cannot be altered or changed once deployed on the blockchain. Thus, smart contract development allows substituting traditional contracts and develops business growth in several industries, namely supply chain management, logistics and shipping, insurance, and charity. In this category, we discuss cross-organizational collaboration-driven smart contract usage solutions, which can be categorized into profit-centric solutions (see Table 10, Table 11, Table 12) and non-profit-centric solutions (see Table 13).

8.1 Profit-centric solutions

The smart contract protocol aims at making contracts more secure, executed in real-time, and more transparent, which are the exact challenges with the existing profit-centric cross-organizational collaboration. Profit-centric solutions aim at increasing the profit by reducing real-time tracking costs, improving cross-border payments, and enhancing distributed problem-solving transparency. Table 10, 11, and 12 present some examples of profit-centric solutions concerning tracking-based solutions, digital asset-based solutions, and crowdsourcing-related solutions, respectively.

8.1.1 Tracking-based solutions

Although business processes may operate well within a centralized mechanism managing internal activities with individual local databases, there still exists a demand for transparency across processes and trust relationships among involved parties. Indeed, real-time tracking may reduce the unnecessary wait for the confirmation of information. Thus, using a distributed system can enhance the transparency and performance of business processes. Smart contracts can be used to automate the transfer of various types of ownership of assets, property, and value and therefore, lead to more visible and less-intermediated working processes. In this context, several studies using smart contracts have been proposed for supply chain management of foods [11][17][53], manufactured products [24][43][50][102], shipped items [39], bio-drugs [105], and imported products [108]. For instance, Casado-Vara et al. [17] have proposed a model for agriculture tracking involving blockchain, smart contracts, and a multi-agent system. The blockchain is used to store all transaction information in the supply chain. Besides, the multi-agent system used smart contracts to manage the entire supply chain process more efficiently while removing intermediaries. Furthermore, according to industry estimations, the global halal

Table 10: Some examples of profit-centric solutions: Tracking-based

Publication	**Contribution**	**Description**
[11]	Supply chain system	It is a generic agri-food supply chain traceability system based on blockchain technology implementing the "from-farm-to-fork" (F2F) model currently used in the European Union, which can integrate current traceability rules and processes.
[17]	Supply chain model	It is based on blockchain that aims at coordinating the tracking of food in the agriculture supply chain using smart contracts and a multi-agent system.
[24]	Granularity level framework	It is a generic framework for defining granularity levels based on the product's unique characteristics, supply chain processes, and stakeholders' engagement by using smart contracts within a blockchain-enabled supply chain traceability architecture.
[39]	Supply chain management	It is a blockchain-based solution for efficient supply chain management involving items shipped via smart containers. Smart contracts are used to manage shipment conditions, automate payments, legitimize receivers, and also issue a refund in case of violations to pre-defined conditions.
[43]	IC Traceability method	It is a method of integrated circuit (IC) supply chain traceability based on blockchain. Smart contracts allow supply chain participants to authenticate, track, trace, analyze, and provision chips throughout their entire life cycle.
[50]	Makerchain	Makerchain is a decentralized blockchain-driven model to handle the cyber-credit of social manufacturing among various makers. Smart contracts are used to automate the verification of the product life-cycle through a trail of historic events.
[53]	Food Traceability system	It is a trusted, self-organized, open, and ecological food traceability system based on blockchain and Internet of Things technologies.
[102]	Tracing manufacturing processes	It is a system that allows for traceability of manufactured goods, including their components using tokens.
[105]	QuarkChain	QuarkChain is a blockchain-enabled interoperability framework and it has the reputation based Proof-of-Authority as a preliminary smart contract design for addressing challenges in biopharmaceutical supply chain management.
[108]	originChain	It is a blockchain-based traceability system that provides transparent tamper-proof traceability data with high availability and enables automated regulatory-compliance checking and adaptation in imported product traceability scenarios.

food market will reach USD 2.55 trillion by 2024 [92]. Thus, several companies are using blockchain to improve traceability in the halal food supply chain. For instance, a UK based company has partnered with a blockchain platform provider in order to track livestock and fresh food from farm to table through the halal food chain using the blockchain technology [92].

8.1.2 Digital asset-based solutions

Because of their resilience to tampering, smart contracts are appealing in many scenarios, especially in those which require transfers of money to respect certain agreed rules like in financial services. Therefore, smart contracts in the finance application domain manage, gather, and/or distribute the money as a preeminent feature. The lack of a centralized authority reduced costs and in theory provided more control and access to the investors [46]. To this end, some smart contracts are used for cross-border payments without relying on banks. For instance, the blockchain payment provider, called Ripple is a blockchain solution for payments that is proven in the real world by connecting existing bank ledgers to facilitate near real-time cross-border payments. Ripple may also reduce costs and provide additional pricing transparency of real-time cross-border payments [3]. Table 11 presents other smart contracts that implemented data/good trading service [7][65][106], insurance service [9], rent/exchange good service [14][28], energy trading and demand management service [55][100], social credit system [107], and mobile payment system [110]. For instance, smart contracts are exploited in the insurance industry to automate claims processing, verification, and payment, thus to increase the speed of claim processing as well as to prevent fraud and reduce manual mistakes. Recently, a smart contract-based flight insurance system has been proposed to refund automatically the insured passengers in case of a flight delay [13]. Moreover, blockchain-based systems can provide solutions to the cyber insurance challenges by realizing an automated, real-time, and immutable feedback loop between the insurer, its customer, and potential auditors [20]. Moreover, blockchain technology can mitigate the problems faced by traditional insurance while complying with religious principles [67]. Indeed, a smart insurance model based on Islamic insurance, called Takaful is proposed in [64]. The main difference between Takaful and conventional insurance that in Takaful, insured funds belong to them, the insurance company is just a manager. Thus, by using blockchain and smart contract technologies, insurance companies can be more transparent, which is the highest feature requested by customers. The authors in [64] have suggested transforming the traditional insurance policies into smart contracts that can be executed automatically in order to refund the policyholders without causing compensations for fake incidents.

8.1.3 Crowdsourcing-related solutions

Crowdsourcing is an online, distributed problem-solving and production model in which individuals or organizations obtain goods and services from a large group of participants. For instance, crowdfunding has become one popular form of collective funding among several categories of crowdsourcing. Crowdfunding is a process, in which small donations or investments, made by groups of people, support the development of new projects in exchange for free products or different types of recognition. Traditional crowdsourcing is based on a central system where requesters post tasks on a central server or platform, however,

Table 11: Some examples of profit-centric solutions: Digital asset-based

Publication	Contribution	Description
[7]	Escrow trade protocol	It is a dual-deposit escrow trade protocol that uses double-sided payment deposits in conjunction with simple cryptographic primitives for provably cheat-proof delivery and payment for a digital good without a trusted mediator based on blockchain-enabled smart contracts.
[9]	CAIPY	CAIPY is a smart contract-based ecosystem for simple and transparent car insurance in which smart contracts do not replace but support current processes to enable significant cost savings for insurance claims.
[14]	DAPP	DAPP is a Decentralised App for the sharing of everyday objects based on a smart contract that enables users to register and rent devices without involvement of a Trusted Third Party (TTP), disclosure of any personal information, or prior sign up to the service.
[28]	FairSwap	It is a protocol for a fair exchange of digital goods using smart contracts that take the role of an external judge that completes the exchange in case of disagreement.
[55]	EV power trading model	Electric vehicles power trading model is based on smart contracts and aims at realizing the information equivalence and transparent openness of power trading.
[65]	IoT data trading marketplace	It is a decentralized, trusted, transparent, and open architecture for IoT traffic metering and contract compliance.
[100]	Energy demand management	It is a hierarchical framework for the energy demand-side management through peer-to-peer exchange of information and energy in the real-time market using smart contracts.
[106]	Data Trading Mode	It is a solution to the data trading mode based on the smart contract using blockchain and machine learning. Smart contracts are used to authenticate and authorize the data owner before authorizing the data purchaser to download the purchased data.
[107]	BLESS	BLESS is a BLockchain-Enabled Social credits System that rewards the residents who commit to socially beneficial activities. Smart contract enabled authentication and authorization strategy prevents any unauthorized entity from accessing the credit system.
[110]	Mobile Payment Scheme	It is a robust mobile payment scheme based on sturdy certificate-less signatures with bilinear pairing while making it suitable for computation-constrained mobile devices.

this centralized model currently faces various challenges such as prohibitive cost, single point of failure, and vulnerability to malicious attacks. To this end, blockchain is considered as a promising technology that aims at addressing the aforementioned challenges by eliminating the single point of failure, enhancing transparency, and enforcing rules using smart contracts. In this context, several studies using blockchain-enabled smart contracts [38][87][97][119] have been proposed, as shown in Table 12. For instance, Zichichi et al. [119] have proposed a smart contract-based social decentralized autonomous organization

Table 12: Some examples of profit-centric solutions: Crowdsourcing-related

Publication	Contribution	Description
[38]	Fluid	Fluid is a blockchain based framework which supports foundations of general crowdsourcing platforms using smart contracts.
[87]	MPCSToken	MPCSToken is a smart contract enabled fault-tolerant incentivisation for mobile P2P crowd services to facilitate service auction, task execution and payment settlement process.
[97]	LoC	LoC is a financial loan management system based on smart contracts over permissioned blockchain Hyperledger Fabric.
[119]	LikeStarter	It is a smart-contract based social decentralized autonomous organization that combines social interactions with crowdfunding mechanisms, allowing any user to raise funds while becoming popular in the social network.

for crowdfunding, called LikeStarter where social network site users can raise funds for other users through a simple "like", built on top of the Ethereum blockchain. Smart contracts are used to control and manage funds without the need for a trusted third entity. LikeStarter assigns Likoins (i.e. tokens related to an artist) to users that fund a given project. These tokens can be employed and converted to buy artifacts and they provide users with voting capabilities (i.e. they can contribute to the decision of the price of certain artifacts).

8.2 Non-profit-centric solutions

Blockchain technology is needed in a cross-organizational collaboration area suffering from a decline in trust from involved parties (e.g., volunteers, donors, voters, etc.) who are unable to know how their contributions are spent/handled. Indeed, smart contracts enable "fully auditable" performance data, which is secure and extremely difficult to falsify or hack. Table 13 presents some examples of non-profit-centric solutions including volunteer system [19], philanthropic-related systems [32][82][91][94], e-voting service [75], system for educational institutions [86], and copyright protection [115]. For instance, Cheng et al. [19] have proposed VOLTimebank, a volunteer time bank system for a mutual pension based on blockchain and smart contracts. VOLTimebank provides a channel for volunteers to serve the elderly and gives volunteers a way to exchange the services they can offer today with the services that they hope to get in the future. In the philanthropy context, the collection processes are not transparent, and due to this, the involved organizations struggle to gain donors' trust and interest. Thus, some efforts have been made to map the charity collection process on blockchain technology, for instance, Farooq et al. [32] have proposed a charity collection platform, which is based on blockchain technology, and is transparent for donors and legal authorities to conduct an audit. The design uses smart contracts and digital wallets to transfer money in real-time with complete data security and an auditable trail of every transaction.

Table 13: Some examples of non-profit-centric solutions

Publication	Contribution	Description
[19]	VOLTimebank	Volunteer time bank (VOLTimebank) is a system for a mutual pension based on blockchain and smart contracts.
[32]	Charity management platform	It is a blockchain-based charity management platform that aims at providing a transparent, secure, auditable, and efficient system. Smart contracts are used to buy, sell, and transfer CharityCoin to organizations and individuals, and call for donations.
[75]	Borda Count Voting	It is a self-tallying decentralized e-voting protocol for a ranked-choice voting system based on Borda count.
[82]	Tracking donation platform	It offers transparent accounting of operations donors, charitable foundations, and recipients based on blockchain technology.
[86]	Computational System	It is a decentralized model of a computational system built on blockchain for educational institutions by introducing a cryptocurrency within the network of the institute.
[91]	Charity-Chain	It is a decentralized network for tracking donations and helping donors (philanthropic organizations, impact investors, small donors) to monitor their transactions and hence restore their trust in giving to such social organizations.
[94]	Smart Donations	It is a blockchain-powered mobile platform and application that facilitates a novel model for real-time, condition-based donations using smart contracts.
[115]	BMCProtector	BMCProtector is a blockchain and smart contract-based application to protect music copyright and ensure holders' income rights.

These smart contracts have been introduced to securely transfer donations to individual beneficiaries, organization, and their associated projects. Zhao and O'Mahony [115] have proposed BMCProtector, a prototype implementation based on an Ethereum blockchain and smart contract technologies, for effective protection of music copyright and rights of copyright owners. The deployed smart contract is responsible for sharing the copyright parameters.

Despite the benefits of blockchain and smart contracts in reforming operations in a wide variety of industries, namely supply chain, insurance, and charity, certain challenges to their widespread adoption still exist. These challenges include legal issues, lack of standards and protocols, privacy issues, and error intolerance. Arguments that smart contracts are no panacea for all financial use cases doubt the applicability of smart contracts to certain scenarios as far as agreement type and scale.

9 Discussion

We discuss below the study results and present challenges and future development trends in smart contract research.

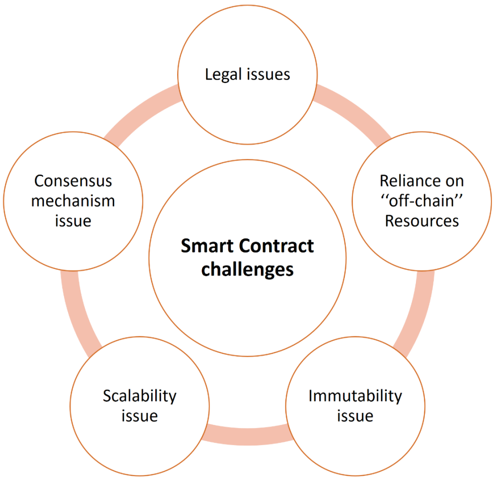

Legal issues
Reliance on "off-chain" Resources
Immutability issue
Scalability issue
Consensus mechanism issue
Smart Contract challenges

Fig. 4: Challenges and Open issues

9.1 Challenges and Open issues

As an emerging technology, smart contracts currently face many challenges, such as legal, reliance on "off-chain" Resources, immutability, scalability, and consensus mechanism issues (see Figure 4).

9.1.1 Legal issues

The legal issue of smart contracts is another crucial aspect of smart contract challenges. For example, the European General Data Protection Regulation (GDPR) [35] stipulates that citizens have a "right to be forgotten" which is inconsistent with the immutable nature of blockchain-enabled smart contracts. Other legal issues can be cited including, (i) each country has its own laws and regulations, hence, it is complicated to ensure compliance will all regulations, (ii) law clauses or conditions are not quantifiable, thus it is still complicated to model these conditions in smart contracts so that they are appropriate and quantifiable for a machine to execute them, and (iii) governments are interested in a regulated and controlled use of the blockchain technology in many applications, however, this means that the untrustworthy network will regress to a third-party trusted network, losing part of its essence [79].

9.1.2 Reliance on "off-chain" Resources

Several smart contracts require receiving information or parameters from resources that are not on the blockchain itself, so-called off-chain resources. For this purpose, oracles are used as trusted third parties that retrieve off-chain information and then push that information to the blockchain at predetermined times. Although existing oracles are well tested, their use may introduce a potential "point of failure". For instance, an oracle might be unable to push out the necessary information, provide erroneous data, or go out of business. Therefore, smart contracts will need to account for these eventualities before their adoption can become more widespread [51].

9.1.3 Immutability issue

The immutability feature is an important characteristic of smart contracts. Indeed, once a smart contract is deployed, the code cannot be changed by any party. However, the dark side of the immutability concept in smart contracts lies mainly in the fact that in the event of any errors made in the code, the immutability feature of a smart contract prevents it from being rectified. Similarly, if circumstances change (e.g., the parties have mutually agreed to change the parameters of their business deal, or if there is a change in law, etc.), no simple path to amend a smart contract is possible. Therefore, extensive and possibly expensive reviews of the smart contract performed by experts before its deployment in a blockchain are required to address the immutability issue.

Another limitation in the blockchain itself that impacts the smart contracts is the irreversible nature of the blockchain, thus once the smart contracts are deployed, they cannot be changed. Moreover, any blockchain nodes can be hacked or misused to report erroneous data that will be logged on the blockchain in an immutable manner.

9.1.4 Scalability issue

Scalability is the primary concern for many blockchain networks. For instance, the Ethereum blockchain can verify 14 transactions per second, which is slow as compared with Visa that can handle up to 24,000 transactions per second. Indeed, the scalability issue leads to network congestion, increased commission fees for transactions, and an increase in the time required to confirm the transactions [80]. In order to address the scalability issue, extensive research focusing on increasing the number of transactions per second by smart contract platforms is required in the future. However, the transaction verification depends on the consensus mechanism used by the smart contract platforms. Therefore, scalability depends on consensus mechanisms, which is another issue in smart contracts.

9.1.5 Consensus mechanism issue

The consensus mechanism plays the leading role to maintain security, scalability, and decentralization in the blockchain networks at the same time. There are several existing consensus algorithms, including Proof-of-Work (PoW), Proof-of-Stake (PoS), etc. Although the PoW algorithm enables security in the blockchain, it wastes resources. Thus, many organizations switch from the PoW algorithm to new consensus mechanisms that promise lower fees for transactions as well as lower energy costs for the block production process. Therefore, future studies can use new consensus mechanisms, such as proof-of-activity (PoA) or delegated proof-of-stake (DPoS) in order to test them and eventually improve their quality.

9.2 Future Development Trends

Future development trends of smart contracts are introduced from two aspects namely, Layer 2 protocols, and contract management solutions.

9.2.1 Layer 2 protocols

In order to address the aforementioned challenges faced by smart contracts, a viable solution, called Layer 2 is appeared to tackle the blockchain scalability problem. While Layer 1 is the used term to describe the underlying main blockchain architecture, Layer 2 is an overlaying network that lies on top of the underlying blockchain. Indeed, Layer 2 refers to the multiple solutions or protocols being built on top of an existing blockchain system. The main goal of Layer 2 protocols is to solve the transaction speed and scaling difficulties that are being faced by the major cryptocurrency networks. Therefore, Layer 2 protocols refer to a secondary framework, where blockchain transactions and processes can take place independently of Layer 1 ("main-chain"). Two major examples of Layer 2 solutions are the Bitcoin Lightning Network [27] and the Ethereum Plasma [76]. The Lightning Network, which in part developed at the MIT Media Lab's Digital Currency Initiative, is a lightweight software solution for scaling public blockchains and cryptocurrency interoperability. It aims at greatly reducing cost and time constraints by shifting small transactions to a cryptographically secure "off-chain" environment so that only large netting transactions need to be directly settled into a resource-constrained blockchain [27]. Ethereum Plasma is a series of smart contracts, which allows for many blockchains within a root blockchain. The root blockchain enforces the state in the Plasma chain. The root chain is the enforcer of all computation globally but is only computed and penalized if there is proof of fraud. Many Plasma blockchains can co-exist with their business logic and smart contract terms. Indeed, Plasma enables persistently operating decentralized applications at a high scale [76]. To sum up, thanks to Layer 2, a great portion of the work that would be performed by the "main-chain" can be moved to the second layer. So while the "main-chain" provides security, the second layer protocols provides better solutions for the scalability issue by offering high throughput, being able to perform hundreds, or even thousands, of transactions per second.

9.2.2 Contract management solutions

Smart contracts encompass far more than just the benefits of blockchain technology. Rather, the term captures the entire digital life cycle of a contract, from negotiation to control and verification of the fulfillment of contractual obligations. Now, it is already possible to use smart contracts even without blockchain technology. Thus, contract management solutions could overcome both the immutability issue and the irreversible nature of blockchain by handling the contract's life-cycle while eliminating limitations of the technology itself. In state-of-the-art contract management solutions [31], all parties to the contract must provide proof of identity and authenticate their access to data in order to ensure the basis of trust. Besides, all documents that are associated with the contract are stored in a revision-secure manner and encrypted form on a cloud-based platform developed and operated in Europe. This ensures transparency and traceability for all events, the actions associated with these events, and the designation of the persons responsible [23]. For instance, Fabasoft Contracts [31] is one of the latest contract management solutions that is ready-to-use, cloud-based software to support users throughout the entire contract life cycle: from cross-company contract preparation, efficient handling of review and approval processes, to the revision-secure contract archiving. It enables the modeling of contract rights and obligations, which can be automatically verified and enforced. There are several benefits offered by revision-secure contract management, including providing traceability when monitoring the cold chain of food delivery or proving the authenticity of spare automotive parts, as opposed to counterfeit articles [23].

10 Conclusion

The decentralization, auto-enforcing ability, and verifiability characteristics of smart contracts enable their encoded business rules to be executed in a peer-to-peer network, where each node is "equal" and none has any special authority without the involvement of a trusted authority or a central server. Thus, smart contracts are expected to revolutionize many traditional industries, such as financial, healthcare, energy, etc. In this paper, we presented a comprehensive survey of blockchain-enabled smart contracts from both technical and usage points of view. Thus, we introduced a taxonomy of existing blockchain-enabled smart contract solutions, categorized the included research papers, and discussed the existing smart contract-based studies. Based on the findings from the survey, both smart contract challenges and open issues are identified to be addressed in further studies. Finally, we discussed future trends of smart contracts. This study provides informational support to stakeholders interested in the research of smart contracts.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Alharby, M., Aldweesh, A., van Moorsel, A.: Blockchain-based smart contracts: A systematic mapping study of academic research (2018). In: 2018 International Conference on Cloud Computing, Big Data and Blockchain (ICCBB), pp. 1–6. IEEE (2018)
2. Amani, S., Bégel, M., Bortin, M., Staples, M.: Towards verifying ethereum smart contract bytecode in isabelle/hol. In: Proceedings of the 7th ACM SIGPLAN International Conference on Certified Programs and Proofs, pp. 66–77. Association for Computing Machinery, New York, NY, USA (2018)
3. Analytics, T.C.: Ripple xrp continue to revolutionize cross border payment systems. Available online at `https://thecurrencyanalytics.com/11696/ripple-xrp-continue-to-revolutionize-cross-border-payment-systems` (2020). Last accessed: 2020-10-03
4. Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., et al.: Hyperledger fabric: a distributed operating system for permissioned blockchains. In: Proceedings of the Thirteenth EuroSys Conference, p. 30. ACM (2018)
5. Angelo, M.D., Salzer, G.: A survey of tools for analyzing ethereum smart contracts. In: IEEE International Conference on Decentralized Applications and Infrastructures, DAPPCON 2019, Newark, CA, USA, April 4-9, 2019, pp. 69–78. IEEE (2019)
6. Angrish, A., Craver, B., Hasan, M., Starly, B.: A case study for blockchain in manufacturing: "fabrec": A prototype for peer-to-peer network of manufacturing nodes. Procedia Manufacturing **26**, 1180–1192 (2018). 46th SME North American Manufacturing Research Conference, NAMRC 46, Texas, USA
7. Asgaonkar, A., Krishnamachari, B.: Solving the buyer and seller's dilemma: A dual-deposit escrow smart contract for provably cheat-proof delivery and payment for a digital good without a trusted mediator. In: 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC) (2019)
8. Atzei, N., Bartoletti, M., Cimoli, T.: A survey of attacks on ethereum smart contracts (sok). In: International conference on principles of security and trust, pp. 164–186. Springer (2017)
9. Bader, L., Bürger, J.C., Matzutt, R., Wehrle, K.: Smart contract-based car insurance policies. In: 2018 IEEE Globecom Workshops (GC Wkshps), pp. 1–7 (2018)
10. Bai, X., Cheng, Z., Duan, Z., Hu, K.: Formal modeling and verification of smart contracts. In: Proceedings of the 2018 7th International Conference on Software and Computer Applications, pp. 322–326. Association for Computing Machinery, New York, NY, USA (2018)
11. Baralla, G., Pinna, A., Corrias, G.: Ensure traceability in european food supply chain by using a blockchain system. In: Proceedings of the 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain, pp. 40–47. IEEE Press (2019)
12. Bartoletti, M., Pompianu, L.: An empirical analysis of smart contracts: platforms, applications, and design patterns. In: International conference on financial cryptography and data security, pp. 494–509. Springer (2017)
13. Bertini, Thomas and Butkute, Kristina and Canessa, Francesco: Smart flight insurance—insureth. Available online at `http://mkvd.s3.amazonaws.com/apps/InsurEth.pdf` (2015). Last accessed: 2020-10-10
14. Bogner, A., Chanson, M., Meeuw, A.: A decentralised sharing app running a smart contract on the ethereum blockchain. In: Proceedings of the 6th International Conference on the Internet of Things, pp. 177–178. Association for Computing Machinery, New York, NY, USA (2016)
15. Bragagnolo, S., Rocha, H., Denker, M., Ducasse, S.: Smartinspect: solidity smart contract inspector. In: 2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE), pp. 9–18 (2018)
16. Buterin, V., et al.: A next-generation smart contract and decentralized application platform. white paper (2014)
17. Casado-Vara, R., Prieto, J., [la Prieta], F.D., Corchado, J.M.: How blockchain improves the supply chain: case study alimentary supply chain. Procedia Computer Science **134**,

393–398 (2018). The 15th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2018) / The 13th International Conference on Future Networks and Communications (FNC-2018) / Affiliated Workshops

18. Chen, T., Li, Z., Zhou, H., Chen, J., Luo, X., Li, X., Zhang, X.: Towards saving money in using smart contracts. In: Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging Results, pp. 81–84. Association for Computing Machinery, New York, NY, USA (2018)
19. Cheng, S., Shi, W., Zhang, H.: Voltimebank: A volunteer system for mutual pension based on blockchain. In: Proceedings of the 2019 International Conference on Blockchain Technology, pp. 75–79. Association for Computing Machinery, New York, NY, USA (2019)
20. Ciocarlie, G., Eldefrawy, K., Lepoint, T.: Blockcis—a blockchain-based cyber insurance system. In: Proceedings of the 2018 IEEE International Conference on Cloud Engineering (IC2E), Orlando, FL, USA, pp. 17–20 (2018)
21. Cuccuru, P.: Beyond bitcoin: an early overview on smart contracts. I. J. Law and Information Technology **25**(3), 179–195 (2017)
22. Dagher, G.G., Mohler, J., Milojkovic, M., Marella, P.B.: Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology. Sustainable Cities and Society **39**, 283–297 (2018)
23. Dangl, A.: Top trends 2020: Hyperautomation and smart contracts. Available online at `https://www.fabasoft.com/en/news/blog/top-trends-2020-hyperautomation-and-smart-contracts` (2019). Last accessed: 2020-10-07
24. Dasaklis, T.K., Casino, F., Patsakis, C.: Defining granularity levels for supply chain traceability based on iot and blockchain. In: Proceedings of the International Conference on Omni-Layer Intelligent Systems, pp. 184–190. Association for Computing Machinery, New York, NY, USA (2019)
25. DHarz, D., Knottenbelt, W.: Towards safer smart contracts: A survey of languages and verification methods. arXiv preprint arXiv:1809.09805 (2018)
26. Dickerson, T., Gazzillo, P., Herlihy, M., Koskinen, E.: Adding concurrency to smart contracts. Distributed Computing pp. 1–17 (2019)
27. Dryja, Tadge and Glasbergen, Gert-Jaap and Lovejoy, James: Layer 2 - the lightning network. Available online at `https://dci.mit.edu/lightning-network/` (2019). Last accessed: 2020-10-20
28. Dziembowski, S., Eckey, L., Faust, S.: Fairswap: How to fairly exchange digital goods. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, pp. 967–984. Association for Computing Machinery, New York, NY, USA (2018)
29. Ellul, J., Pace, G.J.: Alkylvm: A virtual machine for smart contract blockchain connected internet of things. In: 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), pp. 1–4 (2018)
30. Eltayieb, N., Elhabob, R., Hassan, A., Li, F.: A blockchain-based attribute-based signcryption scheme to secure data sharing in the cloud. Journal of Systems Architecture **102**, 101653 (2020)
31. Fabasoft: Fabasoft contracts. Available online at `https://www.fabasoft.com/en/products/fabasoft-contracts` (2020). Last accessed: 2020-10-07
32. Farooq, M.S., Khan, M., Abid, A.: A framework to make charity collection transparent and auditable using blockchain technology. Computers & Electrical Engineering **83**, 106588–106604 (2020)
33. Feng, X., Wang, Q., Zhu, X., Wen, S.: Bug searching in smart contract. arXiv preprint arXiv:1905.00799 (2019)
34. Gao, Z., Xu, L., Chen, L., Shah, N., Lu, Y., Shi, W.: Scalable blockchain based smart contract execution. In: 2017 IEEE 23rd International Conference on Parallel and Distributed Systems (ICPADS), pp. 352–359 (2017)
35. GDPR: Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46. Official Journal of the European Union (OJ) **59**, 1–88 (2016)

36. Guo, H., Meamari, E., Shen, C.C.: Multi-authority attribute-based access control with smart contract. In: Proceedings of the 2019 International Conference on Blockchain Technology, pp. 6–11. Association for Computing Machinery, New York, NY, USA (2019)
37. Gupta, R., Tanwar, S., Al-Turjman, F., Italiya, P., Nauman, A., Kim, S.W.: Smart contract privacy protection using ai in cyber-physical systems: tools, techniques and challenges. IEEE Access **8**, 24746–24772 (2020)
38. Han, S., Xu, Z., Zeng, Y., Chen, L.: Fluid: A blockchain based framework for crowdsourcing. In: Proceedings of the 2019 International Conference on Management of Data, pp. 1921–1924. Association for Computing Machinery, New York, NY, USA (2019)
39. Hasan, H., AlHadhrami, E., AlDhaheri, A., Salah, K., Jayaraman, R.: Smart contract-based approach for efficient shipment management. Computers & Industrial Engineering **136**, 149–159 (2019)
40. Hu, V.C., Ferraiolo, D., Kuhn, R., Friedman, A.R., Lang, A.J., Cogdell, M.M., Schnitzer, A., Sandlin, K., Miller, R., Scarfone, K., et al.: Guide to attribute based access control (abac) definition and considerations (draft). NIST special publication **800**(162) (2013)
41. Hu, Y., Liyanage, M., Mansoor, A., Thilakarathna, K., Jourjon, G., Seneviratne, A.: Blockchain-based smart contracts-applications and challenges. arXiv preprint arXiv:1810.04699 (2018)
42. Hwang, G.H., Tien, P.C., Tang, Y.H.: Blockchain-based automatic indemnification mechanism based on proof of violation for cloud storage services. In: Proceedings of the 2020 The 2nd International Conference on Blockchain Technology, pp. 90–94. Association for Computing Machinery, New York, NY, USA (2020)
43. Islam, M.N., Kundu, S.: Enabling ic traceability via blockchain pegged to embedded puf. ACM Trans. Des. Autom. Electron. Syst. **24**(3) (2019)
44. Javaid, U., Aman, M.N., Sikdar, B.: Blockpro: Blockchain based data provenance and integrity for secure iot environments. In: Proceedings of the 1st Workshop on Blockchain-Enabled Networked Sensor Systems, pp. 13–18. Association for Computing Machinery, New York, NY, USA (2018)
45. Javaid, U., Siang, A.K., Aman, M.N., Sikdar, B.: Mitigating lot device based ddos attacks using blockchain. In: Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems, pp. 71–76. Association for Computing Machinery, New York, NY, USA (2018)
46. Jentzsch, C.: Decentralized autonomous organization to automate governance. White paper, November (2016)
47. Jiang, B., Liu, Y., Chan, W.K.: Contractfuzzer: Fuzzing smart contracts for vulnerability detection. In: Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, pp. 259–269. Association for Computing Machinery, New York, NY, USA (2018)
48. Kolluri, A., Nikolic, I., Sergey, I., Hobor, A., Saxena, P.: Exploiting the laws of order in smart contracts. In: Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis, pp. 363–373. Association for Computing Machinery, New York, NY, USA (2019)
49. Kosba, A., Miller, A., Shi, E., Wen, Z., Papamanthou, C.: Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In: 2016 IEEE symposium on security and privacy (SP), pp. 839–858. IEEE (2016)
50. Leng, J., Jiang, P., Xu, K., Liu, Q., Zhao, J.L., Bian, Y., Shi, R.: Makerchain: A blockchain with chemical signature for self-organizing process in social manufacturing. Journal of Cleaner Production **234**, 7676778 (2019)
51. Levi, S.D., Lipton, A.B.: An introduction to smart contracts and their potential and inherent limitations. In: Harvard Law School Forum on Corporate Governance & Financial Regulation (2018)
52. Lewis, A.: A gentle introduction to smart contracts. Available online at `https://bitsonblocks.net/2016/02/01/gentle-introduction-smart-contracts/` (2016). Last accessed: 2020-10-07

53. Lin, J., Shen, Z., Zhang, A., Chai, Y.: Blockchain and iot based food traceability for smart agriculture. In: Proceedings of the 3rd International Conference on Crowd Science and Engineering. Association for Computing Machinery, New York, NY, USA (2018)
54. Liu, C., Liu, H., Cao, Z., Chen, Z., Chen, B., Roscoe, B.: Reguard: Finding reentrancy bugs in smart contracts. In: Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings, pp. 65–68. Association for Computing Machinery, New York, NY, USA (2018)
55. Liu, H., Zhang, Y., Zheng, S., Li, Y.: Electric vehicle power trading mechanism based on blockchain and smart contract in v2g network. IEEE Access **7**, 160546–160558 (2019)
56. Liu, J., Liu, Z.: A survey on security verification of blockchain smart contracts. IEEE Access **7**, 77894–77904 (2019)
57. Liu, X., Muhammad, K., Lloret, J., Chen, Y.W., Yuan, S.M.: Elastic and cost-effective data carrier architecture for smart contract in blockchain. Future Generation Computer Systems **100**, 590–599 (2019)
58. Loukil, F., Ghedira-Guegan, C., Boukadi, K., Benharkat, A.N.: Towards an end-to-end iot data privacy-preserving framework using blockchain technology. In: International Conference on Web Information Systems Engineering, pp. 68–78. Springer (2018)
59. Luu, L., Chu, D.H., Olickel, H., Saxena, P., Hobor, A.: Making smart contracts smarter. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 254–269. Association for Computing Machinery, New York, NY, USA (2016)
60. Macrinici, D., Cartofeanu, C., Gao, S.: Smart contract applications within blockchain technology: A systematic mapping study. Telematics and Informatics **35**(8), 2337–2354 (2018)
61. Maesa, D.D.F., Mori, P., Ricci, L.: A blockchain based approach for the definition of auditable access control systems. Computers & Security **84**, 93–119 (2019)
62. Memoria, F.: Smartbillions challenges hackers with 1,500 ether reward, gets hacked and pulls most of it out. Available online at `https://www.ccn.com/smartbillions-challenges-hackers-1500-ether-reward-gets-hacked-pulls/` (2017). Last accessed: 2020-10-20
63. Meng, W., Wang, J., Wang, X., Liu, J., Yu, Z., Li, J., Zhao, Y., Chow, S.S.: Position paper on blockchain technology: Smart contract and applications. In: International Conference on Network and System Security, pp. 474–483. Springer (2018)
64. Meskini, F., Islamic, R.A.: Multi-agent based simulation of a smart insurance using blockchain technology. In: 2019 Third International Conference on Intelligent Computing in Data Sciences (ICDS), pp. 1–6. IEEE (2019)
65. Missier, P., Bajoudah, S., Capossele, A., Gaglione, A., Nati, M.: Mind my value: A decentralized infrastructure for fair and trusted iot data trading. In: Proceedings of the Seventh International Conference on the Internet of Things. Association for Computing Machinery, New York, NY, USA (2017)
66. Mohanta, B.K., Panda, S.S., Jena, D.: An overview of smart contract and use cases in blockchain technology. In: 9th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2018, Bengaluru, India, July 10-12, 2018, pp. 1–4. IEEE (2018)
67. Muneeza, A., Arshad, N.A., Arifin, A.T., et al.: The application of blockchain technology in crowdfunding: towards financial inclusion via technology. International journal of management and applied research **5**(2), 82–98 (2018)
68. Murray, Y., Anisi, D.A.: Survey of formal verification methods for smart contracts on blockchain. In: 10th IFIP International Conference on New Technologies, Mobility and Security, NTMS 2019, Canary Islands, Spain, June 24-26, 2019, pp. 1–6. IEEE (2019)
69. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. Available online at `https://bitcoin.org/bitcoin.pdf` (2008). Last accessed: 2020-10-20
70. Nizamuddin, N., Salah, K., Azad], M.A., Arshad, J., Rehman, M.: Decentralized document version control using ethereum blockchain and ipfs. Computers & Electrical Engineering **76**, 183–197 (2019)
71. Nxt community: Nxt whitepaper. Available online at `https://nxtdocs.jelurida.com/Nxt_Whitepaper` (2016). Last accessed: 2020-10-07

72. Omar, A.A., Bhuiyan, M.Z.A., Basu, A., Kiyomoto, S., Rahman, M.S.: Privacy-friendly platform for healthcare data in cloud based on blockchain environment. Future Generation Computer Systems **95**, 511–521 (2019)
73. Osterland, T., Rose, T.: Model checking smart contracts for ethereum. Pervasive and Mobile Computing **63**, 101129 (2020)
74. Ouaddah, A., Elkalam, A.A., Ouahman, A.A.: Harnessing the power of blockchain technology to solve iot security & privacy issues. In: Proceedings of the Second International Conference on Internet of Things, Data and Cloud Computing, ICC'17. Association for Computing Machinery, New York, NY, USA (2017)
75. Panja, S., Bag, S., Hao, F., Roy, B.: A smart contract system for decentralized borda count voting. IEEE Transactions on Engineering Management **67**(4), 1323–1339 (2020)
76. Poon, J., Buterin, V.: Plasma: Scalable autonomous smart contracts pp. 283–295 (2017)
77. Praitheeshan, P., Pan, L., Yu, J., Liu, J., Doss, R.: Security analysis methods on ethereum smart contract vulnerabilities: a survey. arXiv preprint arXiv:1908.08605 (2019)
78. Regnath, E., Steinhorst, S.: Smaconat: Smart contracts in natural language. In: 2018 Forum on Specification & Design Languages (FDL), pp. 5–16. IEEE (2018)
79. Reyna, A., Martín, C., Chen, J., Soler, E., Díaz, M.: On blockchain and its integration with iot. challenges and opportunities. Future generation computer systems **88**, 173–190 (2018)
80. Rieth, Yulia: Payment systems: Visa vs. bitcoin. Available online at `https://decenter.org/en/payment-systems-visa-vs-bitcoin` (2018). Last accessed: 2020-10-10
81. Rouhani, S., Deters, R.: Security, performance, and applications of smart contracts: A systematic survey. IEEE Access **7**, 50759–50779 (2019)
82. Saleh, H., Avdoshin, S., Dzhonov, A.: Platform for tracking donations of charitable foundations based on blockchain technology. In: 2019 Actual Problems of Systems and Software Engineering (APSSE), pp. 182–187. IEEE (2019)
83. Schrans, F., Eisenbach, S., Drossopoulou, S.: Writing safe smart contracts in flint. In: Conference Companion of the 2nd International Conference on Art, Science, and Engineering of Programming, pp. 218–219 (2018)
84. Scoca, V., Uriarte, R.B., De Nicola, R.: Smart contract negotiation in cloud computing. In: 2017 IEEE 10th International Conference on Cloud Computing (CLOUD), pp. 592–599 (2017)
85. Sergey, I., Nagaraj, V., Johannsen, J., Kumar, A., Trunov, A., Hao, K.C.G.: Safer smart contract programming with scilla. Proc. ACM Program. Lang. **3**(OOPSLA) (2019)
86. Shariar, A., Imran, M.A., Paul, P., Rahman, A.: A decentralized computational system built on blockchain for educational institutions. In: Proceedings of the International Conference on Computing Advancements, ICCA 2020. Association for Computing Machinery, New York, NY, USA (2020)
87. Shi, F., Qin, Z., Wu, D., McCann, J.: Mpcstoken: Smart contract enabled fault-tolerant incentivisation for mobile p2p crowd services. In: 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), pp. 961–971 (2018)
88. Shi, N., Tan, L., Li, W., Qi, X., Yu, K.: A blockchain-empowered aaa scheme in the large-scale hetnet. Digital Communications and Networks (2020)
89. Singh, A., Parizi, R.M., Zhang, Q., Choo, K.K.R., Dehghantanha, A.: Blockchain smart contracts formalization: Approaches and challenges to address vulnerabilities. Computers & Security **88**, 101654 (2020)
90. Singla, V., Malav, I.K., Kaur, J., Kalra, S.: Develop leave application using blockchain smart contract. In: 2019 11th International Conference on Communication Systems Networks (COMSNETS), pp. 547–549 (2019)
91. Sirisha, N.S., Agarwal, T., Monde, R., Yadav, R., Hande, R.: Proposed solution for trackable donations using blockchain. In: 2019 International Conference on Nascent Technologies in Engineering (ICNTE), pp. 1–5. IEEE (2019)
92. TE-FOOD: Halal food companies are going to blockchain. Available online at `https://cointelegraph.com/press-releases/halal-food-companies-are-going-to-blockchain` (2018). Last accessed: 2020-10-01
93. team, T.: Truffle: Ethereum development framework. Available online at `https://github.com/trufflesuite/truffle` (2016). Last accessed: 2020-10-20

94. Trotter, L., Harding, M., Elsden, C., Davies, N., Speed, C.: A mobile platform for event-driven donations using smart contracts. In: Proceedings of the 21st International Workshop on Mobile Computing Systems and Applications, p. 108. Association for Computing Machinery, New York, NY, USA (2020)
95. Udokwu, C., Kormiltsyn, A., Thangalimodzi, K., Norta, A.: The state of the art for blockchain-enabled smart-contract applications in the organization. In: 2018 Ivannikov Ispras Open Conference (ISPRAS), pp. 137–144. IEEE (2018)
96. Wang, E.K., Liang, Z., Chen, C.M., Kumari, S., Khan, M.K.: Porx: A reputation incentive scheme for blockchain consensus of iiot. Future Generation Computer Systems **102**, 140–151 (2020)
97. Wang, H., Guo, C., Cheng, S.: Loc — a new financial loan management system based on smart contracts. Future Generation Computer Systems **100**, 648–655 (2019)
98. Wang, P., Liu, X., Chen, J., Zhan, Y., Jin, Z.: Qos-aware service composition using blockchain-based smart contracts. In: Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings, pp. 296–297. Association for Computing Machinery, New York, NY, USA (2018)
99. Wang, S., Yuan, Y., Wang, X., Li, J., Qin, R., Wang, F.: An overview of smart contract: Architecture, applications, and future trends. In: 2018 IEEE Intelligent Vehicles Symposium, IV 2018, Changshu, Suzhou, China, June 26-30, 2018, pp. 108–113. IEEE (2018)
100. Wang, X., Yang, W., Noor, S., Chen, C., Guo, M., [van Dam], K.H.: Blockchain-based smart contract for energy demand management. Energy Procedia **158**, 2719–2724 (2019). Innovative Solutions for Energy Transitions
101. Watanabe, H., Fujimura, S., Nakadaira, A., Miyazaki, Y., Akutsu, A., Kishigami, J.J.: Blockchain contract: A complete consensus using blockchain. In: 2015 IEEE 4th global conference on consumer electronics (GCCE), pp. 577–578. IEEE (2015)
102. Westerkamp, M., Victor, F., Küpper, A.: Tracing manufacturing processes using blockchain-based token compositions. Digital Communications and Networks (2019)
103. WIRED: A 50 million hack just showed that the DAO was all too human. Available online at https://www.wired.com/2016/06/50-million-hack-just-showed-dao-human/ (2016). Last accessed: 2020-10-20
104. Wright, K., Martinez, M., Chadha, U., Krishnamachari, B.: Smartedge: A smart contract for edge computing. In: 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pp. 1685–1690 (2018)
105. Xie, W., Wang, B., Ye, Z., Wu, W., You, J., Zhou, Q.: Simulation-based blockchain design to secure biopharmaceutical supply chain. In: Proceedings of the Winter Simulation Conference, pp. 797–808. IEEE Press (2019)
106. Xiong, W., Xiong, L.: Smart contract based data trading mode using blockchain and machine learning. IEEE Access **7**, 102331–102344 (2019)
107. Xu, R., Lin, X., Dong, Q., Chen, Y.: Constructing trustworthy and safe communities on a blockchain-enabled social credits system. In: Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, pp. 449–453. Association for Computing Machinery, New York, NY, USA (2018)
108. Xu, X., Lu, Q., Liu, Y., Zhu, L., Yao, H., Vasilakos, A.V.: Designing blockchain-based applications a case study for imported product traceability. Future Generation Computer Systems **92**, 399–406 (2019)
109. Yang, Z., Lei, H.: Fether: An extensible definitional interpreter for smart-contract verifications in coq. IEEE Access **7**, 37770–37791 (2019)
110. Yeh, K., Su, C., Hou, J., Chiu, W., Chen, C.: A robust mobile payment scheme with smart contract-based transaction repository. IEEE Access **6**, 59394–59404 (2018)
111. Yu, K., Tan, L., Shang, X., Huang, J., Srivastava, G., Chatterjee, P.: Efficient and privacy-preserving medical research support platform against covid-19: A blockchain-based approach. IEEE Consumer Electronics Magazine (2020)
112. Yu, K.P., Tan, L., Aloqaily, M., Yang, H., Jararweh, Y.: Blockchain-enhanced data sharing with traceable and direct revocation in iiot. IEEE Transactions on Industrial Informatics (2021)

113. Zhang, F., Cecchetti, E., Croman, K., Juels, A., Shi, E.: Town crier: An authenticated data feed for smart contracts. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 270–282. Association for Computing Machinery, New York, NY, USA (2016)
114. Zhang, Y., Kasahara, S., Shen, Y., Jiang, X., Wan, J.: Smart contract-based access control for the internet of things. IEEE Internet of Things Journal **6**(2), 1594–1605 (2019)
115. Zhao, S., O'Mahony, D.: Bmcprotector: A blockchain and smart contract based application for music copyright protection. In: Proceedings of the 2018 International Conference on Blockchain Technology and Application, pp. 1–5. Association for Computing Machinery, New York, NY, USA (2018)
116. Zhao, Y., Liu, Y., Tian, A., Yu, Y., Du, X.: Blockchain based privacy-preserving software updates with proof-of-delivery for internet of things. Journal of Parallel and Distributed Computing **132**, 141–149 (2019)
117. Zheng, Z., Xie, S., Dai, H.N., Chen, W., Chen, X., Weng, J., Imran, M.: An overview on smart contracts: Challenges, advances and platforms. Future Generation Computer Systems **105**, 475–491 (2020)
118. Zhou, H., de Laat, C., Zhao, Z.: Trustworthy cloud service level agreement enforcement with blockchain based smart contract. In: 2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), pp. 255–260 (2018)
119. Zichichi, M., Contu, M., Ferretti, S., DAngelo, G.: Likestarter: a smart-contract based social dao for crowdfunding. In: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 313–318 (2019)
120. Zou, W., Lo, D., Kochhar, P.S., Le, X.B.D., Xia, X., Feng, Y., Chen, Z., Xu, B.: Smart contract development: Challenges and opportunities. IEEE Transactions on Software Engineering (2019)