IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 17, 2020, accepted March 12, 2021, date of publication March 23, 2021, date of current version June 24, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3068178

Survey on Blockchain-Based Smart Contracts: Technical Aspects and Future Research

THARAKA MAWANANE HEWA[1], (Student Member, IEEE), YINING HU[2], (Student Member, IEEE), MADHUSANKA LIYANAGE[1,3], (Senior Member, IEEE), SALIL S. KANHARE[4], (Senior Member, IEEE), AND MIKA YLIANTTILA[1], (Senior Member, IEEE)

[1]Centre for Wireless Communication, University of Oulu, 90570 Oulu, Finland
[2]IBM Australia, Melbourne, VIC 3006, Australia
[3]School of Computer Science, University College Dublin, Dublin 4, Ireland
[4]Department of Computer Science and Engineering, University of New South Wales, Sydney, NSW 2052, Australia

Corresponding author: Tharaka Mawanane Hewa (tharaka.hewa@oulu.fi)

This work was supported by the 5GEAR Menot CWC-NS under Grant 2430299111, and in part by the framework of 6Genesis Flagship under Grant 318927.

ABSTRACT The industrial and computing research context revolutionized in various directions during the last decades. The blockchain-based smart contract embraced as a significant research interest due to its distinguishing features such as decentralized storage of transactions, autonomous execution of contract codes, and decentralized establishment of the trust. Blockchain-based smart contracts can transform the working architecture of almost all industries towards elevated service standards. The use cases of blockchain based smart contracts range from industrial applications such as cryptocurrency systems towards logistics, agriculture, real estate, energy trading and so forth. The decentralization concept of blockchain is one of the biggest leaps in technology research since future computing got a super momentum towards the Internet of Things (IoT) and edge computing. A plethora of research is in progress to investigate the opportunities for the applicability of smart contracts and blockchain technologies to various industries. It is important to identify the technical aspects of blockchain-based smart contracts to further improve and sharpen the capabilities which they already owed. This survey is conducted to identify the significant technical aspects of blockchain-based smart contracts with the associated future research directions.

INDEX TERMS Blockchain, concurrency, Ethereum, Hyperledger Fabric, IoT, smart contracts, security, scalability.

I. INTRODUCTION

The computation associated services including financial transactions, video streaming, email, and telecommunication mostly adopted a centralized architecture [1] with the influence of design principles such as cloud computing. Mail servers, video streaming servers as well as payment authorization systems have been deployed in centralized computing infrastructure and users consume these services adopt the client-server architecture. However, the future evolutionary directions of computing research reflect that the number of subscribers of computational services expands enormously [2]–[4]. The Industry 4.0 revolution has anticipated the contribution of industrial IoT, Machine to Machine communication and Artificial Intelligence (AI). The centralized systems have scalability limitations with the future massive demand expansions of computing. Therefore, the computing research community is highly motivated to investigate a novel architectural strategy for sustaining in the computational demand for the next generation [5]. In addition, privacy, availability, access control, and data integrity are major concerns in computing research. Blockchain-based smart contracts augment their employability as a technology breakthrough to the future industry by built-in decentralization with integrity, autonomous execution and accuracy [6].

The blockchain is a decentralized, immutable ledger which is composed of a cryptographically linked chain of blocked records. The collection of records is called blocks and the records are usually called transactions or events.

The associate editor coordinating the review of this manuscript and approving it for publication was Junaid Arshad.

The decentralized ledger is shared within all contributory members in the blockchain network [7]. Transactions are added to the ledger upon verification and agreement process between the parties on-board in the blockchain. The cryptographic link is the backbone of blockchain. The important keywords associated with blockchain are the ***decentralization***, ***immutability*** and ***cryptographic link***.

Decentralization: The decentralization reflects the transactional(store and retrieve data) capability of blockchain based smart contracts without a single point of failure. The ledger is available on each node and in contrast with centralized database management systems [8], the access to the data does not depend on a centralized service.

Immutability: The records in the ledger are immutable once logged [9]. The attempt to forge the ledger record on a particular block will disqualify it and fail the data integrity of the entire blockchain. The immutability of the ledger ensured using cryptographic techniques such as hashing and digital signatures which will explain in the paper. The alterations of ledger is a computationally expensive task.

Cryptographic Link: The cryptographic link is the backbone of trust of the entire blockchain [10]. The immutability of blockchain is achieved through the cryptographic link established with hashing and digital signatures [11]. Neither the transaction or block can be altered since it requires altering all subsequent blocks.

A. PAPER MOTIVATION

Blockchain-based smart contracts have an immense application context, ranging from various financial applications [18]–[24], to health-care applications [25]–[32]. Correctly programming smart contracts, however, has been shown to be challenging. For example, a financial loss on the Ethereum network [33] was caused by bad programming practices.

There exist several surveys on blockchain-based smart contracts. Wright *et al.* [34] present the benefits and drawbacks of the emerging decentralized technology and its requirement to the expansion of a new subset of law that termed as Lex Cryptographia and highlighted the requirement of the regulation of blockchain-based smart contract based organizations under legal theory. Wüst *et al.* [35] critically analyze the applicability of blockchain for a particular application scenario by proposing a structured methodology to determine the relevant technical solutions and evaluated with some significant real-world applications. Clack *et al.* [36] explored the design landscape of potential formats for storage and transmission of smart legal agreements in association with blockchain technology specifically for the financial services context. Wang *et al.* [37] also provides a comprehensive overview of blockchain-powered smart contracts highlighting the distinguished challenges in the smart contracts along with future trends. Nonetheless, existing research works have not thoroughly studied the technical aspects of smart contracts. They also have not explored the potential of integrating smart contracts to other technologies, such as artificial intelligence and game theory.

B. OUR CONTRIBUTION

In this paper we aim to conduct a through survey focusing on several technical aspects of smart contracts. More specifically, we discuss issues on the security, privacy, gas cost, concurrency of existing smart contract programming languages and make the following contributions:

- We first discuss existing issues and existing solutions on the security, privacy, gas cost and concurrency of smart contracts.
- We then discuss the lessons learned and future research directions for the development of smart contracts to improve their security, privacy, gas cost and concurrency.
- We also discuss future research topics that involve integrating smart contracts with other technologies including artificial intelligence and game theory.

C. OUTLINE OF THE PAPER

The paper consists of five sections. SectionI provides a brief introduction to the paper. Table 1 consists of a few important surveys conducted in the smart contract context previously. Section II consists of significant prerequisites for understanding smart contracts. Important concepts of cryptography, including hashing, digital signatures are explained. The principles of blockchain and its core modules are also discussed. In addition, the evolution of blockchain towards smart contracts along with important historical milestones are examined in the paper. Section III holds significant portion of the core contribution of this survey. Table 2 contains the key components of some leading smart contract platforms in the market. Section III includes the significant technical aspects of the blockchain-based smart contracts. The content includes the security attacks, precautions and best practices to eliminate these attacks, performance optimization and scalability improvement techniques. The significant insights from the technical aspects of the blockchain-based smart contract and important considerations for shaping the future research directions were elaborated in Section IV. Other future research directions, such as combining smart contracts with game theory and artificial intelligence are later discussed in Section V. Finally, Section VI concludes the survey.

II. BACKGROUND

The concept of smart contracts was first introduced by Nick Szabo. Ethereum [20] is one of the most prominent smart contract platforms with multitudinous applications in different contexts. Initially, smart contracts were targeted only for financial applications such as ERC20 Tokens [38]. Over time, the invention of smart contract platforms diversified due to various industrial requirements [39].

TABLE 1. Previous surveys on smart contracts.

Ref	Year	Description	Comparison with our contribution
[12]	2019	**Security and Privacy on Blockchain**: A comprehensive overview on security and privacy of blockchain, with techniques to enforce security and privacy of smart contacts.	We extend the discussion on security and privacy further and discuss the additional technical aspects such as performance optimization, concurrency, scalability and formal verification.
[13]	2019	**Applications of Distributed Ledger Technologies to the Internet of Things: A Survey**: Provides an analysis on the blockchain in the application perspective.	Our paper has focused the technical aspects of utilizing of smart contracts on different applications including IoTs.
[14]	2019	**Inter Blockchain Communication: A Survey**: A comprehensive discussion on cross blockchain communication techniques.	Our paper discusses several relevant technical aspects, not limiting to one specific topic.
[15]	2018	**A Survey of Blockchain Applications in Different Domains** : Provides an overview on the application contexts of blockchain in domains including currency, healthcare, copyright protection, and insurance.	Our paper has mainly focused the technical aspects of utilizing smart contracts in different applications.
[16]	2018	**Understanding the Software Development Practices of Blockchain Projects: A Survey**: Includes results of a formal survey to identify the software engineering practices including requirement analysis, task assignment, testing, and verification of blockchain software projects.	We discuss the technical aspects rather than the application perspective.
[17]	2018	**A Survey on Opportunities and Challenges of Blockchain Technology Adoption for Revolutionary Innovation**: Discussed the blockchain adoption across the significant sectors in the Industry 4.0.	We discuss the technical aspects in general rather than focus on single applications.

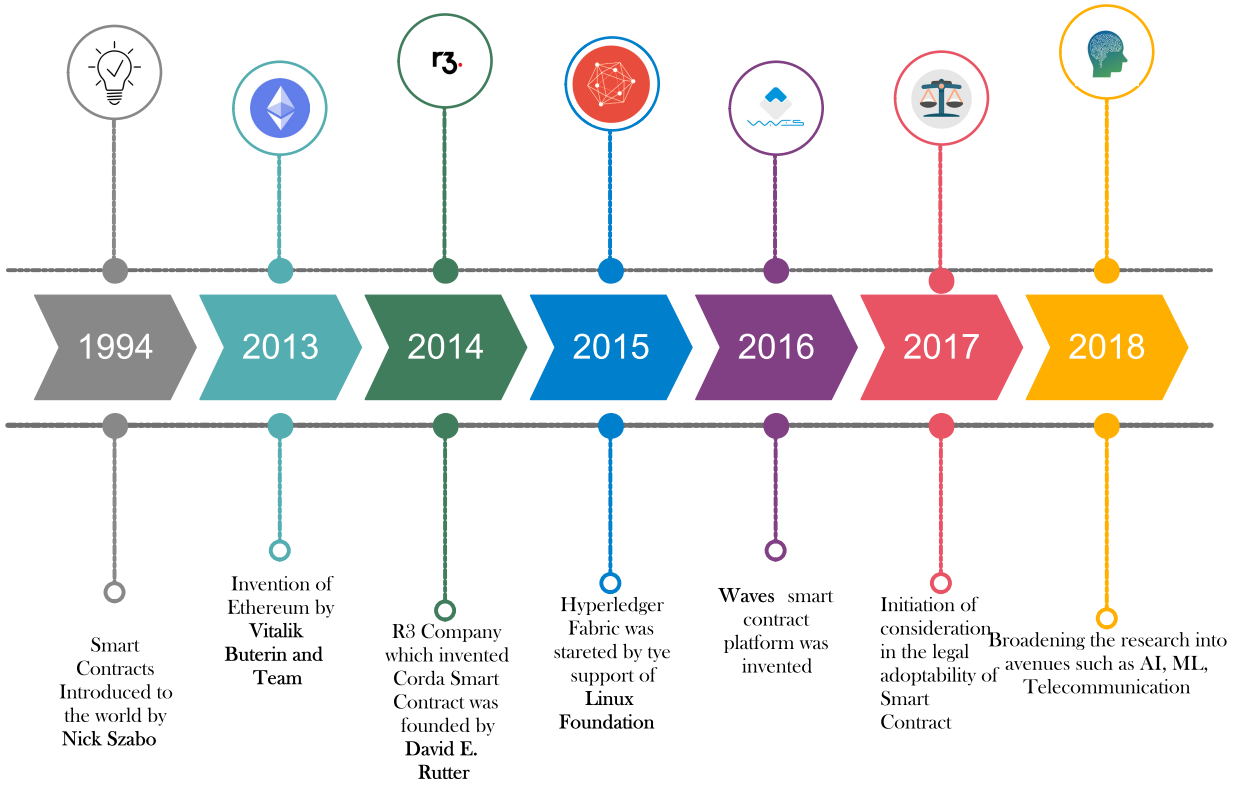


FIGURE 1. Evolution timeline of significant blockchain platforms [40]–[43].

A. HISTORY OF SMART CONTRACTS

Figure 1 illustrates the important milestones of the historical evolution of blockchain-based smart contracts. The introduction of the concept of smart contracts was by Nick Szabo to the world in 1994 is the birth of smart contracts. The invention of Ethereum is one of the most important leaps of the smart contract history. The public Ethereum blockchain allowed the users to get on board and deploy smart contract applications in the public blockchains. Ethereum was primarily targeted for currency exchange at the beginning. The Hyperledger Fabric project was initiated in collaboration with the Linux Foundation. The direction of Hyperledger Fabric has deviated from Ethereum's since Hyperledger was intended as an enterprise blockchain. Many platforms being developed target the enterprise requirements. The research focused on the legalization of smart contracts with the maturity of smart contract context with multiple platforms and diverse use cases. The next generation of research is highly focused on the position of smart contracts in the emerging research topics in computer science.

TABLE 2. Some leading smart contract platforms in the market.

Platform	Ref	Description
Ethereum	[20]	Ethereum is a public blockchain platform which enables the users to deploy smart contracts publicly. The computational resource consumption for the execution of the smart contract evaluated and the native cryptocurrency is Ether.
Hyperledger	[42]	Hyperleder Fabric is an open source blockchain platform invented by the Linux foundation and few other organizations. The platform developed as an enterprise blockchain platform and there is no native cryptocurrency attached to the Hyperledger platform.
NEM	[60]	NEM is blockchain platform which designed for the enterprise and enables asset definition which can be mapped as the realistic industrial instruments such as legal items and shipping documents.
R3 Corda	[41]	R3 Corda is a scalable blockchain platform which enables privacy features and designed for the industrial integration.
Stellar	[61]	Stellar is a blockchain based multi-currency payment platform which is comparably efficient in terms of computational resource consumption.
Waves	[43]	The Waves platform is a cryptocurrency platform which also integrated with a dollar payment gateway to make the wallet users capable for the replenishment in US dollars.
Ethereum Classic	[62]	Ethereum Classic was formed by forking Ethereum, which enables transactions for a lower cost. It is evolving as an improved version of Ethereum.
Tezos	[63]	Tezos is defined as a self-amending crypto ledger, with native cryptocurrency and two types of accounts as implicit accounts and originated accounts. The smart contracts are attached to the originated accounts.
NEO	[64]	Neo is developed intending a smart economy. Neo can be used for digitize the assets such as digital certificates.
Cardano	[65]	Cardano smart contract platform is the first platform being backed by the peer-reviewing and scientific study. The Cardano has introduced a new programming language called Plutus to develope smart contracts.
EOS	[66]	EOS is developed mainly targeting to build the horizontally and vertically scalable decentralized applications. EOS has addressed many aspects including parallel processing, governance and improved usability.
Qtum	[67]	Qtum is a smart contract platform which is known as a hybrid of Bitcoin and Ethereum. It is adapted to the UTXO model of Bitcoin and included native cryptocurrency.
Lisk	[68]	Lisk is a smart contract platform that enables Javascript-based decentralized applications. The authors have introduced the concept of side-chain and SDK which can be attached to the blockchain if required.
ARK	[69]	ARK platform was developed with improved consensus and scalability. The authors defined the improved simplicity which allows a user to deploy a new blockchain and a token at the push of a button.
RSK	[70]	RSK is a scalable blockchain which enables near realtime payments with Turing complete smart contract incorporation to the Bitcoin blockchain.
NXT	[71]	NXT provides a template based smart contracts which are non Turing complete. The smart contract templates require to match with the business.

B. FEATURES OF BLOCKCHAIN-BASED SMART CONTRACTS

Smart contracts are self-enforcing and self-executing programs which actuate the terms and conditions of a particular agreement using software codes and computational infrastructure. The smart contracts are decentralised programs that extend the use of the underlying blockchain network [44]. The program is immutable and is cryptographically verified to ensure its trustworthiness.

Some features of smart contracts inherit from the underlying blockchain technology. These features enable the employability of smart contracts across diverse domains [45]. Generally speaking, smart contracts execute in the peer to peer mode without the intervention of a centralized third party. They provide service availability without any centralized dependency. And allow automated transaction execution when pre-defined conditions are reached [46]. Below we detail the key features of blockchain-based smart contracts.

1) ELIMINATION OF TRUSTED THIRD PARTY AND AUTONOMOUS EXECUTION

The most significant advantage of blockchain-based smart contracts is decentralization [47]. The requirement of trusted intermediaries such as brokers, agents or service providers can be dislodged when a particular system integrated with blockchain-based smart contracts. Elimination of a trusted third party will reduce the transaction costs and authority imposed by centralized entities. One of the most significant examples is cryptocurrency which embraced smart contracts to altering the role of trusted third parties such as central banks [48]. The centralized third parties impose high costs for transactions and behave as the ultimate governing bodies. The users need to adhere to the regulations imposed by the centralized authorities.

In contrast, smart contracts provide the agreement procedure to be defined by the participants themselves maximizing democracy [49]. The participants define the rules and regulations for the smart contract establishment and deploy upon mutual agreement. The programmed condition and flow of events are supposed to execute once the blockchain reached a specific pre-defined state. The specific state will be defined in the smart contract upon agreement of all parties in the blockchain network. This state can be any condition such as a specific balance of wallet funds, or a specific time bound, etc. The execution is then automatic without intervening a centralized third party. The service availability is guaranteed since the operation does not rely on a centralized third party and execute peer-to-peer. The autonomous execution as per the conditions ensures the accuracy of operation without human error or even without biased actions. Therefore, the smart contract is a promising solution for most applications which require alternatives without trusted third parties.

2) FORGE RESISTANCE AND IMMUTABILITY

The integrity of the transaction records in distributed ledger is verified with digital signatures [50]. Furthermore, the individual transactions verified and approved prior appending to the ledger. The ever-growing ledger consists of approved transactions which are immutable. The alteration cannot be committed by an individual. Smart contract code deployed

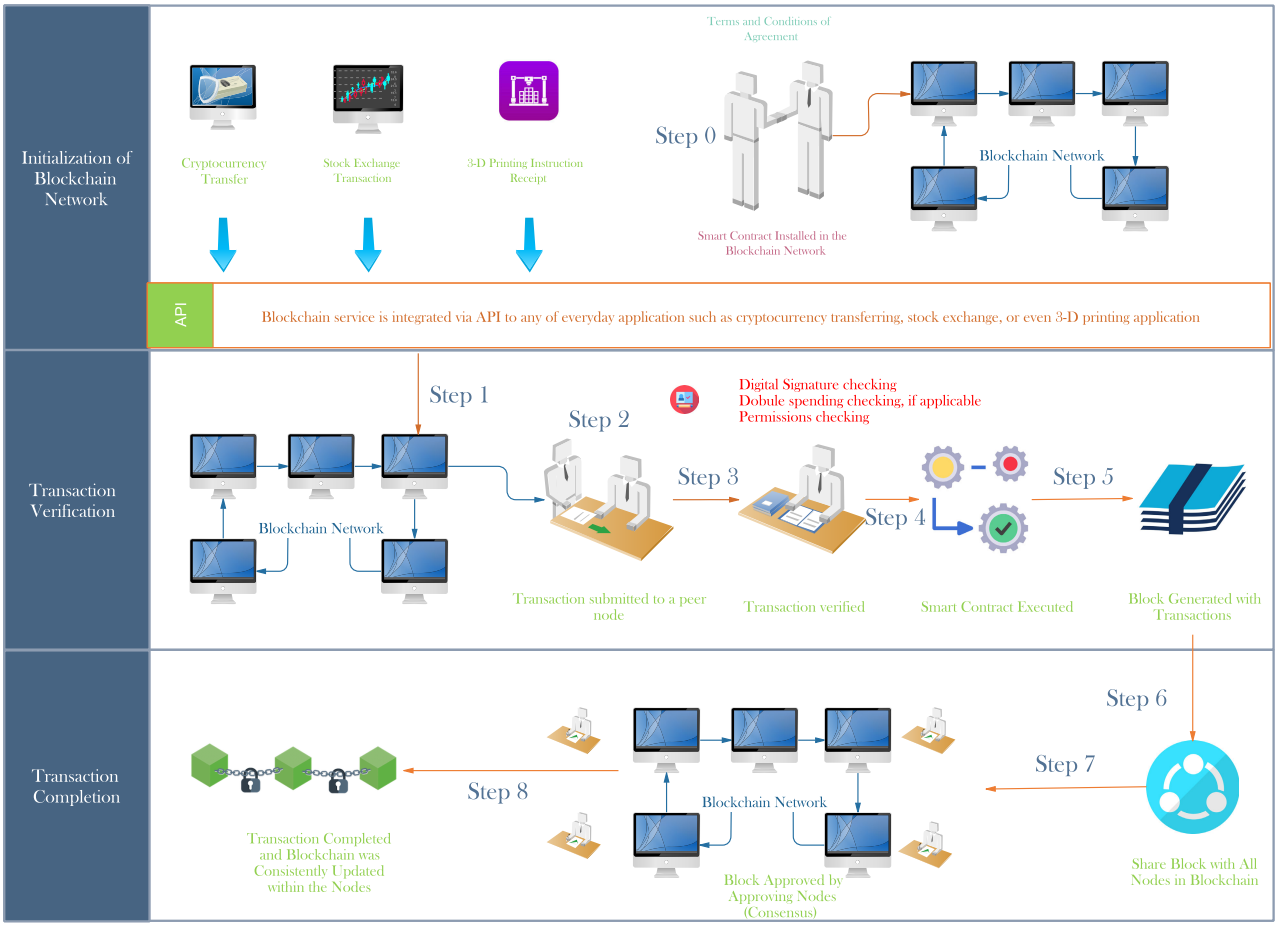


FIGURE 2. High-level transaction flow of a typical blockchain integration.

on the blockchain are immutable. The code can be deployed on each node using various techniques. For instance, as an executable enclosed in the container. The smart contract code is tamper-evident and the tampered smart contracts cannot be executed. However, smart contracts can be updated if required upon the agreement of nodes in the blockchain network. Therefore, all parties in the blockchain network can trust the smart contract and trust that the executed code contains the logics disclosed to and agreed by each member of the blockchain network.

3) TRANSPARENCY

Transparency [51] is one of the significant distinguishing features inherited to the smart contracts from blockchain [52]. The transparency of the smart contract is twofold. Firstly, the code defined in smart contracts is transparent to intervening parties as well as to the public. Secondly, the set of transactions included in the blocks are also transparent to the public. Hence the intervening parties of the blockchain network can trust the logic and transactions in the blockchain network [53]. In a more concrete example, if the smart contract logic defined by a governing authority who is a participant of blockchain network [54], the particular operation executed upon to the logic can be regarded as trusted and unbiased since the code is publicly visible. Furthermore, the transaction added to the ledger is also publicly visible to ensure trust [55]. In contrast, the centralized service architecture is not transparent and is prone to vulnerabilities such as man-in-the-middle attacks. The centralized databases are also vulnerable and impossible to trace if any modification occurred on the data on rest. The smart contract code transparency [56] ensures members of the blockchain ecosystem to publicly verify its execution correctness.

C. DEPLOYMENT AND EXECUTION

Figure 2 portraits significant milestones of the initialization and transaction processing of the blockchain-based smart contracts. The initial step (Step 0) involves initialization of smart contracts. After defining the terms and conditions as a software program, the smart contract requires to be installed on the network. The smart contract deployed in each node is identical in all aspects to ensure the fairness and fulfill the principal requirement of blockchain-based smart contracts. There are many interfacing techniques in the market for each blockchain network, when smart contracts are required to connect with the external business systems. The REST

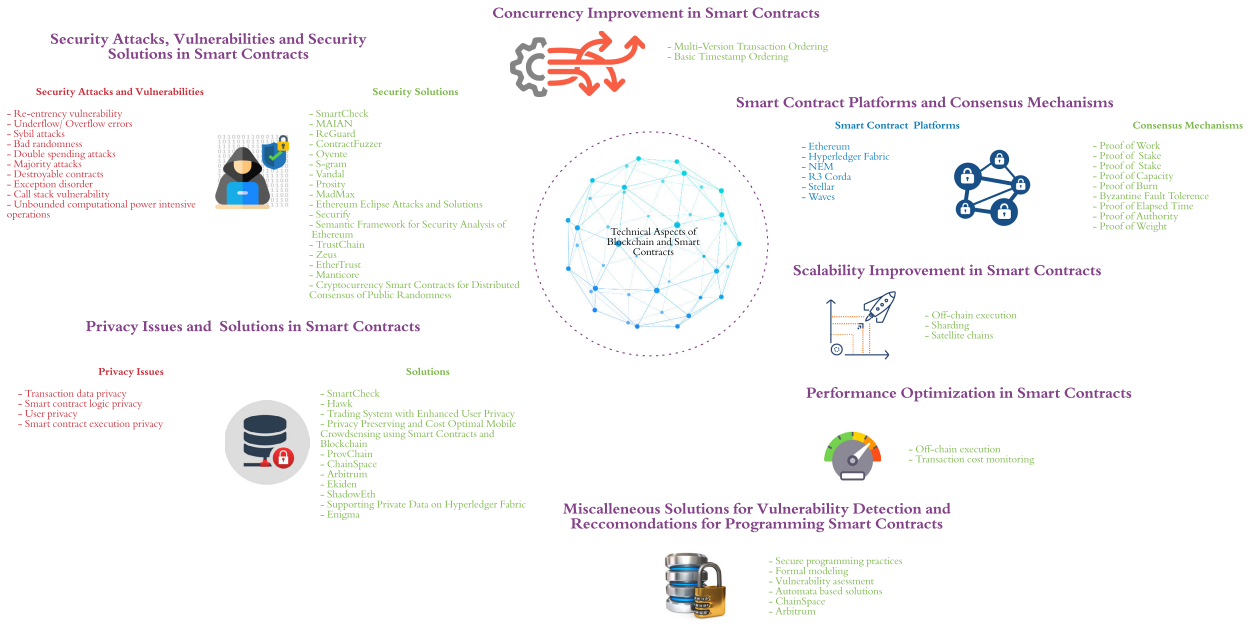


FIGURE 3. Overview of the technical aspects of smart contracts.

API created with Hyperledger Fabric SDK [57] or Ethereum SDK [58] based applications are significant examples. The blockchain network receives transactions from the interfacing applications (Step 1). Once the blockchain network has received the transaction, the transactions are verified for several conditions (Step 2). The digital signature is essential to authenticate that the transaction is legitimate and is aroused by the actual member of the network. Furthermore, there are platform-specific checks in some blockchain networks. For instance, the Bitcoin platform checks for the double-spending in this step [59]. Once the transaction is legitimate, the transaction is flagged (Step 3) as a legitimate transaction, and the smart contract gets executed (Step 4). The transaction is added to the block and the finalized block is generated by a mining node (Step 5). The confirmed block is then disseminated within the network (Step 6), and receives approvals from every node (Step 7) based on the pre-defined consensus rule. Once the consensus condition is met, the block is appended to the blockchain (Step 8).

III. TECHNICAL ASPECTS OF SMART CONTRACTS

The scientific research on blockchain-based smart contracts mesmerized on different directions of the technical specialization. Started as a minimal decentralized immutable ledger, the blockchain technology has emerged with vital features such as improved security, scalability and optimal operation thanks to the contribution of the research conducted worldwide. The significant technical aspects of blockchain-based smart contracts are discussed in this section. Figure 3 shows a quick overview of technical aspects of smart contracts.

A. SECURITY ATTACKS, VULNERABILITIES AND POSSIBLE SOLUTIONS

Smart contracts are beginning to be widely adopted in many industry use cases. Security is of vital importance in consideration of smart contracts as it affects the functionality of entire blockchain. The various attacks on the smart contracts, existing research over these attacks, and potential solutions are discussed in this section. Due to the wide adoption of Ethereum smart contracts in real-life use cases and security attacks occurred in recent years, in this section we focus only on Ethereum to discuss the attacks and vulnerabilities of smart contracts [72]. Table 3 summarizes the attacks and corresponding solutions.

1) DIFFERENT ATTACKS AND VULNERABILITIES

There exist multitudinous attacks in the smart contract context [73]. These attacks lead due to numerous reasons such as programming errors, restrictions in the programming languages, and security loopholes. The results of these attacks consist of many complications for the blockchain networks and their accuracy, loss of native cryptocurrency and terminate the availability of the system. The attacks can be identified on both public and private blockchain platforms. The role of smart contracts and their accuracy can be more momentous in the public blockchains than in private blockchains. The debugging or any correction is a cumbersome process [74] since the contracts adopted to all nodes in a blockchain network.

a: RE-ENTRENCY VULNERABILITY (A1)

In general terms, the re-entrancy vulnerability exploited when one smart contract invokes another smart contract iteratively

TABLE 3. Security attacks and solutions.

Ref	Description	Re-entrency vulnerability (A1)	Underflow/ Overflow errors (A2)	Sybil attacks (A3)	Bad randomness (A4)	Double spending attacks (A5)	Majority attacks (A6)	Destroyable contracts (A7)	Exception disorder (A8)	Call stack vulnerability (A9)	Unbounded computational power intensive operations(A10)
[92]	SmartCheck : Comprehensive analysis tool to detect code issues in Ethereum smart contracts	✓	✓				✓				
[93]	MAIAN : Inter-procedural symbolic analysis and validation of Ethereu,		✓		✓						
[94]	ReGuard : Fuzzy-based analyzer to detect re-entrency vulnerabilites in Ethereum	✓									
[95]	ContractFuzzer : Fuzzy framework to detect vulnerabilities in Ethereum	✓		✓						✓	
[96]	Oyente : A framework to detect bugs in the smart contracts	✓			✓					✓	✓
[97]	S-gram : Prediction tool fo potential vulnerabilities by irregular token sequences	✓									
[98]	Vandal : Security analysis framework for Ethereum with conversion capability of EVM byte code into semnatic relations	✓						✓	✓		
[99]	Prosity : Decompiler for Ethereum which generates readable Solidity syntaxes from bytecode	✓			✓					✓	
[100]	MadMax : Static programming analysis technique to detect gas focused vulnerabilities		✓								✓
[101]	Ethereum Eclipse Attacks and Solutions : Vulnerabilities in consensus and block synchronization and countermeasures					✓					
[102]	Securify : Scalable security analyzer for Ethereum	✓									
[103]	Semantic analysis framework : Semantic security analysis framework for Ethereum using F* programming language	✓							✓		
[77]	TrustChain : Permission-less tamper proof data structure with sybil resistance			✓		✓					
[104]	Zeus : Correctness verifier and fairness validator for smart contracts	✓									
[105]	EtherTrust : Automated static analyzer for EVM bytecode	✓									
[106]	MantiCore : Binary analyzer for Ethereum smart contracts to detect logic bombs	✓									
[107]	Cryptocurrency smart contracts for distributed consensus of public randomness : A secured random number generator				✓						

and the smart contract that initiates the invocation is malicious. Such attacks were ample in the Ethereum platform. Eventually, the invoking smart contract's Ethers are transferred to the malicious contract's account. Mehar *et al.* [33] described the DAO attack, an anonymous hacker stole 50M USD worth of Ethers from the 168M invested through a virtual venture capital raising in 2016. A coder found some loophole, once a split function is invoked, the code retrieves Ether first and update balance later and not checking whether is it a recursive call. The attacker recursively calls split

functions and retrieves their funds before the code checks the balance.

b: UNDERFLOW/OVERFLOW ERRORS (A2)

Underflow or overflow occurred when the result of a particular arithmetic operation was either less or more than the minimum or maximum numeric data type utilized in the smart contract platform. The Ethereum platform is using uint256 [75]. Conceptually, the Ether balance of 0 can be transformed into the maximum value of the uint256 or Ether balance of maximum value can transform into 0. However, the programmers are required to consider such circumstances when coding the contract [76]. There are libraries available to eliminate the known issues, which will discuss in upcoming sections.

c: SYBIL ATTACKS (A3)

Sybil attack [77] occurs when a member attempts to take over the peer network by conceiving fake identities explicitly. Such circumstances [78] will lead to a disproportionate control of the network which can lead to the hijacking of the distributed blockchain peer ecosystem. For instance, if the consensus is based on the majority voting of a particular blockchain, the members explicitly created on the attack can take over the consensus of the network. The system is prone to such a risk when the onboarding process to the blockchain is very minimal and automated.

d: BAD RANDOMNESS (A4)

Randomness is required in the smart contracts, especially in gaming and gambling context. In addition to that, there are utility functions which require randomness [79]. The approaches for random or pseudo-random number generation will be vulnerable in some circumstances [80]. The usage of block variables and block hashes as sources of entropy can be vulnerable.

e: DOUBLE SPENDING ATTACKS (A5)

The spending of the native token is common in almost all smart contract platforms [81]. In each transaction, there is a risk that the same user can spend a particular token two or more times [82]. This type of attacks is known as the double spending attacks [83].

f: MAJORITY ATTACKS (A6)

Majority of attacks occur when some malicious users or groups take over the control to rewrite transaction history or prevent new transactions from confirming [84]. The attack can occur when a particular blockchain consortium adopted with majority voting consensus [85]. Depending on the user requirement, sometimes block mining and transaction verification is offloaded to some dedicated leading peers of each member of the consortium. If the majority of leading peers are hijacked by the malicious user, the similar circumstance occurred.

g: DESTROYABLE CONTRACTS (A7)

The self-destruct vulnerability [86] removes the content of a smart contract by deleting the bytecode at the particular addresses. Furthermore, it sends all contract's funds to a specific target address, making the contract non-functional.

h: EXCEPTION DISORDER (A8)

Exception disorder occurs due to the exceptions lead to a failure when they were not properly handled [87]. Exception handling is an important practice in programming, either blockchain or any other context. The improper handling of exceptions, especially when one contract invokes another, affects the operability of the entire network [88]. Hence, the exception disorder is highlighted as major vulnerability.

i: CALL STACK VULNERABILITY (A9)

The call stack's depth is limited up to a certain value in the execution environment of the smart contract. For instance, in Ethereum the call stack's depth limited to 1024 frames. The operation fails when the depth of the call reaches the limitation. This can occur due to various reasons such as programming errors [89].

j: UNBOUNDED COMPUTATIONAL POWER INTENSIVE OPERATIONS (A10)

Each operation on the smart contracts requires consumption of computational power [90]. For instance, the cost for the computational power in Ethereum is called Gas. The gas utilized to evaluate the computational resource consumption of a particular operation on the smart contract. Unbounded and unrestricted computational power intensive operations lead to various errors and eventually affect the system [91].

2) POTENTIAL SOLUTIONS

Security issues caused by semantic flaws can lead to massive financial loss. Destefanis *et al.* [108] presented a case-study regarding a smart contract library named as Parity. The problem was due to poor programming practices which caused 500,000 Ethers to freeze in 2017.[1] The authors analyzed the chronology of events and identified that the problem occurred due to negligent programming practices.

Since the smart contract programs are deployed in every node of the blockchain system, the requirement of the accuracy of the smart contract is supreme. The smart contract is required to be depleted by vulnerabilities and programming errors before pushing into the thousands of blockchain nodes. There has been a number of research works conducted to address various security attacks over smart contracts. We discuss these research works from the three main categories below: ***identifying semantic flaws***, ***security check tools*** and ***formal verification***.

[1] Equals to 150M USD in year 2017.

a: IDENTIFYING SEMANTIC FLAWS

A number of research studies have analysed these semantic flaws and identified programming practices to mitigate them.

For example, Atzei *et al.* [109] analyzed the vulnerabilities of Ethereum, which is popular in the industry. The vulnerabilities were grouped into three classes according to the level they are introduced, as Solidity, EVM bytecode, or blockchain. The authors highlighted that they expect the non-Turing complete, human readable languages will resolve some of the issues identified.

Delmolino *et al.* [110] documented some important insights from teaching smart contract programming to undergraduate students in the University of Maryland. The authors exposed common errors in designing safe and secure smart contracts, and highlighted the importance of fixing these errors in programming.

Similarly, Wöhrer *et al.* [111] presented several security patterns which are applicable to Solidity developers in order to mitigate typical attack scenarios in Ethereum platform. The patterns declared included protection of re-entrancy attacks, enabling mutexes etc. The authors planned for creating a structured and informative design pattern language for Solidity.

b: SECURITY CHECK TOOLS

Further, multiple security check tools have also been proposed to prevent semantic flaws of smart contracts.

Several studies have tackled the ***re-entry attack***. Liu *et al.* [94] presented the tool ReGuard, which are usable to identify re-entrency bugs in smart contracts. It is a fuzzy-based analyzer which automatically detects the re-entrency bugs in Ethereum smart contracts. ReGuard iteratively generates random diverse transactions to test the vulnerability. Similarly, Jiang *et al.* [95] presented ContractFuzzer, a comprehensive fuzzing framework to detect seven types of vulnerabilities in Ethereum smart contracts. The authors identified few significant types of attacks such as gas-less send and re-entrency vulnerability. The authors identified the false negative rate optimized when comparing with other platforms.

Other research works looked into ***Ethereum bytecode***. For example, Brent *et al.* [98] provided a security analysis framework for Ethereum smart contracts. It provides an analysis pipeline for the conversion of the low-level EVM bytecode into semantic logic relations. The evaluation conveyed that Vandal is fast and robust as well as outperforming leading state-of-art tools with successful analysis of 95 of all 141,000 unique contracts with an average runtime of 4.15 seconds. Suiche *et al.* [99] presented Prosity, a decompiler which generates readable Solidity syntaxes from EVM bytecode. The decompiled contracts can perform with static and dynamic analysis as required. Grishchenko *et al.* [103] later presented a complete small-step semantics of EVM bytecode and formalized a significantly large fragment of EVM using F*, which is a popular programming language used for similar verification programmed proof assistant. The authors also successfully validated it against official Ethereum test suite. The authors further defined number of salient security properties for smart contracts. More recently, Mossberg *et al.* [106] introduced an open source dynamic execution framework named Manticore to analyse the binaries of Ethereum smart contracts. The framework provides analysis to find issues including logic bombs. The API provides flexibility to customize the utilization of framework. This supports scalability up to larger contracts. The authors tested the tool with Oyente and observed outperforming results and EtherTrust showed better precision on a benchmark rather than state-of-art solutions.

As Ethereum contracts consume ***gas***, the gas cost on the smart contract execution has also become a vital concern. GRECH *et al.* [100] classified and identified the gas focused on vulnerabilities found in the Ethereum smart contracts. The gas is the cost of a particular smart contract execution on public Ethereum network and gas-focused vulnerabilities mainly referred to the codes with exhaustive execution to consume allocated gas for the smart contract. In addition to that, the authors presented MadMax, which are some static programming analysis techniques usable to detect gas related vulnerabilities with significantly high confidence. The approach included low-level analysis for decompilation in declerative program analysis techniques for higher level analysis which validated with 6.6 million contracts.

Nikolic' *et al.* [93] implemented MAIAN, which employs an inter-procedural symbolic analysis and concrete validator to identify real exploits. The tool identifies three main types bugs including ***suicidal*** contracts that can kill by anyone, ***prodigal*** contracts that can send Ethers to anyone and greedy contracts which does not allow to get Ether to anyone. The tool evaluated with analysis of one million contracts and flags 34,200 contracts vulnerable spending 10 seconds per contract.

More ***general-purposes*** security check tools were also proposed. For example, Tikhomirov *et al.* [92] proposed SmartCheck, a comprehensive analysis tool that detects security issues of Ethereum smart contracts. The authors evaluated the tool on a massive dataset of real-life contracts and yielded successful results. They also stated the capability of development of the tool in future directions including improvement of grammar. Smartcheck has also been shown to be more effective in automated security testing than other tools [112]. Another example, Tsankov *et al.* [102] presented Securify, a security analyzer for Ethereum smart contract. It is scalable, fully automated and capable of proving the contract behaviors are safe or unsafe corresponding to a given property and tested with more than 18k contracts. The analysis is a two stepped process which includes a symbolic analysis of contract's dependency graph to extract precise semantic information and checking for the compliance violation patterns. Luu *et al.* [96] proposed a symbolic execution tool named as Oyente to find potential security bugs. The tool flagged 8,833 contracts as vulnerable out of the 19,366 including the DAO bug which led to a 60 million USD

loss. Later, Kalra *et al.* [104] also presented the framework Zeus, which can be utilized to verify the correctness and validate the fairness of smart contracts. They also defined the correctness as adherence to safe programming practices and fairness as the adherence to agree upon higher-level business logic. The framework significantly outperforms Oyente with zero false negatives in their data set. Mavridou *et al.* [113] introduced FSolidM, which included a meticulous semantics for designing contracts as finite state machines. The authors presented a tool for creating Finite State Machine (FSM) on a user friendly graphical interface which automatically generates Ethereum smart contracts. The authors also introduced a set of design patterns which can implement as plugins and easily possible to integrate to enhance security and functionality. Liu *et al.* [97] proposed a semantic-aware security auditing technique called S-gram for Ethereum. The authors combined N-gram language modeling as well as lightweight static semantic labelling and to learn statistical regulators of contract tokens and to capture high-level semantics such as the flow sensitivity of a transaction. The authors showed that S-gram is usable to predict potential vulnerabilities in identify irregular token sequences and possible to optimize existing in-depth analyzers.

c: FORMAL VERIFICATION

Formal verification in general refers to formally verifying the correctness of a computer program. Formal verification is important in the context of smart contracts as smart contracts may hold financial values and are often made accessible to everyone on a blockchain. Several research studies have investigated the formal verification of smart contracts, and applied formal verification on different stages during the deployment of smart contracts. For example, Bhagavan *et al.* [114] outlined a framework to analyze and verify runtime safety. While Abdellatif *et al.* [115] and Nehai *et al.* [116] proposed to verify smart contracts in their execution environment. Albert *et al.* [117] on the other hand proposed the EthIR framework to analyze Ethereum bytecode.

B. PRIVACY ISSUES AND SOLUTIONS

Decentralization is a core principle of the blockchain based smart contracts. The decentralization in blockchain makes the transaction ledger and smart contracts transparent to all peers in the network as a feature of security. The transparency is not recommended for certain circumstances. The in-built transparency is a significant privacy concern in blockchain based smart contracts. Table 4 reflects the related works and corresponding privacy solutions.

1) DIFFERENT PRIVACY CONCERNS

Privacy is a broader domain in terms of the smart contract [130]. Due to the distributed nature of smart contracts, there are few major privacy concerns [131]. These privacy concerns [132] need to be addressed in order to increase the employability of smart contracts to the industry [133].

TABLE 4. Privacy issues and solutions.

Ref	Description	Transaction data privacy (I1)	Smart contract logic privacy (I2)	User privacy (I3)	Privacy in execution of smart contracts (I4)
[118]	Hawk	✓		✓	
[119]	Trading system with enhanced user privacy			✓	
[120]	Privacy preserving and cost optimal mobile crowdsensing using smart contracts and blockchain			✓	
[121]	ProvChain			✓	
[122]	ChainSpace	✓			
[123]	Arbitrum			✓	
[124]	Ekiden				✓
[125]	TownCrier				✓
[126]	ShadowEth				✓
[127]	Supporting private data on Hyperledger Fabric with secure multiparty computation	✓			
[128]	Enigma			✓	
[129]	Zether		✓		

a: TRANSACTION DATA PRIVACY (I1)

In certain circumstances [134], members in the network will not prefer transparency since it will reveal some sensitive information such as trade secrets, pricing information. Even though the system associated with blockchain, the required measures for privacy preservation should be integrated [135], [136].

b: SMART CONTRACT LOGIC PRIVACY (I2)

The smart contract publicly deployed in all nodes of the blockchain [129]. Due to some requirements, the business logic of the organization required to be incorporated in the blockchain. The business logics may include sensitive information such as commissions, bonuses. The revealing of such

sensitive information through the smart contract will be a privacy concern of the organizations.

c: USER PRIVACY (I3)

User privacy is highly concerned in some significant applications of blockchain based smart contracts. The users incorporate the smart contracts are required to be private in certain circumstances. For instance, the solutions like health information systems do not prefer by the users if the personal identity information being revealed in the ledger. The privacy of user identity is also a significant concern in the implementation of blockchain based smart contracts [137].

d: PRIVACY IN EXECUTION OF SMART CONTRACTS (I4)

The smart contracts are programs which execute on the computational infrastructure [138]. In blockchain, the smart contracts are executed on the nodes. The instructions executed on the machine can be accessed using multiple approaches [139], [140]. For instance, the password entered by a user required to be loaded into the memory and can be viewed in clear form using memory dump tools. These type of lower level data thefts at the execution are regarded as privacy violations in certain circumstances [141].

2) POTENTIAL SOLUTIONS

We below discuss and categorize various possible solutions on how to preserve smart contract privacy.

a: PRESERVING PRIVACY OF TRANSACTION DATA

Ibáñez *et al.* [142] presented significant insights on different aspects of blockchain technology including general data protection regulation and its applicability on blockchain as an enabler for data protection. The authors discussed application of smart contracts on permissioned blockchains and permissionless blockchains in association with appropriate data controllers. The authors categorized the two types of solutions in enabling compliance, as integration of different cryptographic functions and private computation schemes without revealing contents of transactions and application of blockchains as decentralized verification machines.

Juels *et al.* [143] illustrated the emergence of the criminal smart contracts which will facilitate to reveal the confidential information. The authors illustrated a few issues including theft of cryptographic keys by criminal smart contracts. Their results highlighted creating policies and technical safeguarding measures against criminal smart contracts to ensure the smart contracts' beneficial objectives.

Kosba *et. al* [118] presents Hawk, a privacy preserving smart contracts, which dissipated the privacy hurdle encountered in Bitcoin and Ethereum as a currency. The authors propose a framework, which enables a non specialist programmer to write a privacy preserving smart contract. Hawk guarantees on-chain privacy, which cryptographically hides the flow of money and amount from public's view.

b: PROTECTING USER PRIVACY

Niya *et al.* [119] demonstrated a design and implementation of a trading application which utilized Ethereum smart contracts. The application is developed with flexibility in requesting user identity directly by the seller and the buyer. Lightweight blockchain is established to facilitate data exchange in device-to-device channels.

Chatzopoulos *et al.* [120] proposed a new architecture for the event based spatial crowd-sensing tasks in association with the blockchain and technology with user privacy preservation. The architecture utilizes smart contracts to allow crowd-sensing service providers to submit their requests, run cost optimal auctions and handle payments.

Liang *et al.* [121] designed and implemented ProvChain, which is a decentralized architecture for trusted cloud data provenance. Provchain provides significant security features such as tamper-proof provenance and user privacy. The main operational phases are provenance data collection, provenance data storage and provenance data validation which provides tamper-proof records to enable transparency and data accountability in the cloud.

c: PRIVACY IN THE LOGIC

Al Bassam *et al.* [122] presents ChainSpace, which offers privacy-friendly extensibility in the smart contract platform. The platform offers higher scalability than the existing platform achieved through sharding across nodes using a novel distributed atomic commit protocol named as S-BAC. It also supports auditability and transparency.

Kalodner *et al.* [123] presented Arbitrum, which is a cryptocurrency system with smart contracts. Arbitrum's model is compatible for private smart contracts which does not reveal the internal state to the verifiers who are involved in the validation of transactions in certain circumstances. Arbitrum incentivizes the parties to agree off-chain on the VM's behavior which means that the Arbitrum miners only required to verify digital signatures without revealing the contract to confirm that parties agreed on VM's behavior.

d: TRUSTED EXECUTION ENVIRONMENT (TEE)

Trusted execution environment such as Intel SGX [144] guarantees confidentiality and privacy during code execution.

Zhang *et al.* [125] presented an authenticated data feed system which is named as Town Crier. Town Crier provides a bridge between smart contracts and existing websites which are commonly trusted for non-blockchain applications. The frontend and hardware backend combined with the solution which is enabled with privacy as required.

Cheng et al [124] presented Ekiden, which combines blockchain with TEE. The authors leveraged a novel architecture which separates the consensus from execution and enabled confidentiality preserving smart contracts in trusted execution environment. The authors planned to extend their work to enable secure multi-party computation in future.

Yuan *et al.* [126] presented ShadowEth, which is a system that leverages a hardware enclave to ensure the confidentiality of smart contracts in public blockchain like Ethereum. The system also ensures integrity and availability. The authors implemented the prototype using Intel SGX on Ethereum network to analyse the security and vulnerability of the system.

e: SECURE MULTI-PARTY COMPUTATION

Benhamouda *et al.* [127] presented a method for making Hyperledger Fabric blockchain platform compatible with private data using secure multi-party computation. The protocol implemented utilizing Yao's millionaire problem [145] and oblivious transfer. The authors associated a helper server, which separates multi-party computation into off-chain.

Zyskin *et al.* [128] presented Enigma which is a computational model based on a highly optimized version of secure multi-party computation named as Enigma which guarantees a verifiable secret-sharing schemes and ensure confidentiality. The authors used a modified distributed hash table to hold secret-shared data with an external blockchain as the controller of the network to control the access and identity management. The private components of the smart contracts run off-chain on Enigma platform and named as private contracts.

C. REDUCING COMPUTATIONAL OVERHEADS OF SMART CONTRACTS

The smart contract execution is a resource-intensive process. The stakeholders of the smart contract should compensate for the execution for the relevant parties. In Ethereum, the cost is evaluated in gas. The under-optimal smart contracts are expensive in the computation and eventually will incur an additional cost to the users of smart contracts. Furthermore, the overflown resource consumption of smart contracts will crash the entire system. Therefore, the optimal execution condition is highly anticipated in the context of smart contracts.

1) POTENTIAL SOLUTIONS

Idelberger *et al.* [146] presented important insights with technical advantages and disadvantages of logic-based smart contracts when featuring ordinary contracts. The authors proved that a logic based approach can complement advantageously on its procedural component in few dimensions including negotiation, formation, storage. The authors emphasized that logic and procedural approaches are not incompatible in smart contracts. Chen *et al.* [147] developed GASPER, a tool which locates the Ethereum smart contract gas costly patterns by analyzing the smart contract bytecode. The creators can be overcharged by under-optimized smart contracts by extra gas consumption. In the evaluation, the authors discovered 3 representative predefined patterns from 4,240 smart contracts. Kothapalli *et al.* [148] implemented an incentive compatible protocol called SmartCast which is running off-chain and rely on existing cryptocurrency for the reward and punishment mechanism implementation. The approach created a system which enables the workers to enforce the integrity can reward for their correct participation in the process which were enforced through Ethereum smart contracts. The authors evaluated the feasibility of the approach by building a prototype implementation which comprises Ethereum smart contract and off-chain consensus protocol.

D. CONCURRENCY IMPROVEMENTS

Once smart contracts are deployed on a blockchain, they are expected to execute in multiple instances. To improve efficiency, several studies have proposed approaches to improve concurrency of smart contracts. Two main categories of approaches, including Basic Timestamp Ordering (BTO) and Multi-Version Transaction Ordering (MVTO) were proposed. BTO assigns every transaction a timestamp, and determines the serializability order of transactions for the execution or access to a resource based on the timestamps. MVTO guarantees that if inconsistency is detect between two transactions that access relevant data items, one of them will abort. For instance, Zhang *et al.* [149] proposed a concurrent scheme to run smart contracts in concurrent manner which yielded 2.5 times processing speed in block validation with three working threads. Anjana *et al.* [150] developed an efficient framework based on Software Transactional Memory systems (STMs) to enable concurrent execution of smart contracts. The proposed framework yielded 3.6× and 3.7× speedups over the serial miners under BTO and MVTO respectively.

IV. LESSONS LEARNED AND FUTURE WORK

The previous section reflects the significant technical aspects of the blockchain based smart contracts from a wider perspective. The drawbacks and some solutions to existing smart contracts are analyzed. This section elaborates with significant insights from the technical aspects of smart contracts and research directions for further improvements.

A. SECURITY ATTACKS, VULNERABILITIES AND SECURITY SOLUTIONS IN SMART CONTRACTS

1) LESSONS LEARNED

Ordinary software programming languages can be used in the programming languages used for smart contracts, e.g., Java, Javascript, and GoLang. These languages are designed to be Turing-complete to achieve full functionality. Programming smart contracts will be exposed to human errors as other ordinary software programs. The damages resulting from the programming errors are exponential since the smart contracts are deployed to all nodes. Smart contracts are distinguishing since once the code is deployed, they will be distributed on the entire network which makes it harder to patch as ordinary programs. Therefore, programmers must make sure the smart contract programs are guaranteed to be bug-free. The research on improvement of smart contract errors is evolving and the programmers encouraged to utilize the research outcomes, such as improved libraries. The programmers are capable of using a private network to simulate the attacks or formal

penetration testing for the evaluation of the response in smart contracts for the attack scenario. The smart contracts required to update with the patches on the smart contract security vulnerabilities identified by research and make sure the programs developed by them are free of these known vulnerabilities.

Additionally, the principles of programming are applicable for smart contract programming. Smart contracts are required to program with simplicity to eliminate overheads. The different hardware specifications of the blockchain nodes need to be considered. The integration of loops must be minimized and recursive executions require elimination and development. When the numbers are manipulated, it is essential to prevent the codes from arithmetic overflows. If there are specific libraries existing in order to eliminate such errors, these libraries require to be utilized in the smart contracts. The codes which will lead to deadlocks require identification and elimination before the deployment of the smart contract. Overall, the smart contracts should be designed with consideration of efficiency of memory and computation.

Moreover, as the application areas of smart contracts expand, the codes need to be formally verified. The formal verification requires that the particular smart contract, eventually a computer program executes as per the formal specification anticipated by the stakeholders. The formal specification of smart contracts requires to clearly define with the support of experts. Especially when the underlying blockchains are integrated with mission-critical systems such as air-traffic management and healthcare systems, the formal verification will be a mandatory requirement. The formal verification should not require any third-party intervention. For instance, the vulnerability detection requires expert penetration testers to simulate security attacks and detect vulnerabilities. The security audits may require expert intervention. But the formal verification only requires to establish formal specifications which can be expertise of the developers. The smart contract developers should be aware of formal verification methods in order to verify the smart contracts before deployment. The formal verification identifies significant vulnerabilities such as locking the funds, sending funds to other accounts continuously without the account owner's consent and so on. The formal verification is a mandatory best practice for future smart contract developers.

2) FUTURE WORK

The smart contract programming languages are Turing-complete in most of the leading platforms. Turing completeness leads the entire smart contract system into security risks as per the previous investigations. Therefore, some of the smart contract platforms such as Stellar are designed with Turing incomplete smart contract language. Although Turing incomplete smart contracts do not provide the full functionality as Turing complete ones, some of the security risks are eliminated. Assignment of computational power consumption limitations such as the gas limit will be a prudent development consideration in the smart contracts which will eliminate resource consumption overflows. The correctness evaluation of smart contracts is an essential consideration in the future development of smart contract systems. Re-entrancy attack (A1) was addressed in most of the solutions as per Table 3. Sybil attacks (A3), majority attacks (A6), destroyable contracts (A7) and exception disorders(A8) requires addressing further by upcoming security solutions before attacks leading to financial loss.

In addition to that third party utility libraries requires to develop in parallel. For instance, the on-chain cryptographic operations not supported by the Hyperledger-Fabric platform. If the smart contracts are used as a cryptographic utility, it is required to import cryptographic libraries. It is a vital requirement to check that the third-party libraries imported are free of vulnerabilities. If not, importing the libraries will make the entire blockchain system vulnerable. Syntax improvements over smart contracts will also allow business stakeholders to design the smart contracts of their own, with minimal knowledge of programming. Furthermore, the secure design principles of smart contracts will evolve as recommended design patterns as leading programming languages already defined.

The formal verification of smart contracts is anticipated as a global standard in future smart contracts development. Developers, platform vendors will adopt the formal verification compatibility of the blockchain platforms. There are opportunities for researchers to develop frameworks to conveniently design formal specifications which are the prerequisites for the formal verification of smart contracts. The techniques such as AI can be consolidated into the next generation's formal verification methodologies. In the future, the service platforms for formal verification can also be made available online for popular smart contract platforms. This type of service architecture can be used to support the AI-assisted formal verification of smart contracts.

B. PRIVACY ISSUES AND SOLUTIONS

1) LESSONS LEARNED

The main feature of the distributed ledger technology is the transaction data visibility to all peers of the network. Most people are reluctant to adopt blockchain technologies due to the lack of privacy and transaction data visibility. A plethora of research conducted in the enhancement of privacy in distributed ledger technologies. Privacy enhancement is a mandatory requirement when smart contracts are incorporated into future business systems. Since transparency is a vital strength of blockchain, privacy improvement techniques should not violate the transparency of the blockchain system. One possible solution is to store data off-chain. To incorporate privacy and encryption with smart contracts, a robust key management framework is required. Sometimes the key management framework requires to link with HSMs to align with the compliance requirements. As per Table 4, the smart contract logic privacy requires further attention in the research as privacy enhancement. Moreover, the privacy requirement customizations should also be compliant with the concrete business use case.

2) FUTURE WORK

In the future, smart contracts can be expected to integrate with many business systems with different privacy requirements. Therefore, more privacy enhancement techniques can be anticipated from the research. The distributed and transparent nature of blockchain-based smart contracts can be observed as a contradictory feature from a privacy perspective. However, different techniques will be utilized in future smart contract systems to balance privacy and decentralization. The smart contract key management will also be an emerging direction of research in future blockchain systems. The privacy will be a data security compliance requirement such as PCI-DSS/ PA-DSS for financial systems and HIPAA for healthcare data management systems. If the blockchain-based smart contracts are to be adopted with industry use cases, it is mandatory to design the smart contracts with provisions to align with the regulations. In addition to the transaction data, user privacy is also a vital requirement in blockchain networks. The future blockchain networks are required to be designed for the synergistic operation of existing user management systems. The smart contract systems are required to be compatible with existing PKI based user management solutions along with hardware-assisted authentication schemes, such as smart cards or hardware tokens. Modular architectural design will be an ideal design principle in future blockchain systems to simplify the integration with existing platforms. For secure computation using smart contract data, there also exist various opportunities in applications such as secure multi-party computation.

C. REDUCING COMPUTATIONAL OVERHEADS OF SMART CONTRACTS

1) LESSONS LEARNED

Performance optimization is a major requirement of smart contracts. Smart contracts are often integrated with applications such as financial, aviation, identity management, and access control, where real-time operation with minimal latency and higher throughput are expected. Since the widely used Ethereum blockchain does not support concurrency, there exist limitations of expanding the Ethereum's application domain. The storage service of the ledger, consensus mechanism and smart contract programming languages are the main dependencies of the performance of smart contracts. In addition, transaction verification of the underlying blockchain also affects performance. For instance, the double-spending check is additional verification performed in the financial smart contracts. The smart contract optimization is achievable in different ways. Optimization of consensus protocol is an effective approach. Multi-Version Transaction Ordering and Basic-Timestamp Ordering are some of the consensus optimization techniques. Integration of the Ripple consensus, Stellar smart contract platform reduced transaction processing time into 3-5 seconds. The incorporation of high write throughput also improves the performance of the distributed database. The estimation of gas consumption before the deployment of Ethereum smart contracts ensures that the execution is restricted within the limits when they are deployed on a public blockchain network.

2) FUTURE WORKS

Since smart contracts are deployed on the public blockchain and they affect the efficiency of entire blockchain system, it is important that smart contracts are optimized. Penalization of under optimal smart contracts will be an effective solution to eliminate such under optimal conditions. A global performance rating mechanism for well-known blockchain platforms such as Ethereum can be developed to evaluate independently the performance of smart contracts. Before deploying smart contracts in the blockchain, the performance rating should be approved by the stakeholders. The evaluation can be deployed as a service platform. However, smart contract developers are now biased towards deploying simplified smart contracts and transferring computational overheads to off-chain services. If the computational overheads are transferred off-chain, the data privacy and integrity become vulnerable. Furthermore, the off-chain service can be an additional factor that limits the performance. For off-chain integration, the REST API can be utilized to pass the data to be computed to the outside. However, the REST API will also have certain limitations from a performance perspective. Therefore, alternative integration techniques such as gRPC can be used. If the computational overhead is transferred to Edge nodes in the future, COAP can be used to the integration.

D. CONCURRENCY IMPROVEMENTS

1) LESSONS LEARNED

Concurrency is essential for the smart contracts to cope with the future demands. The smart contract execution and consensus mechanisms should be improved to fulfill the concurrency demands. The concurrency improvements should not trade off security features. The block synchronization and maintenance of the consistency in the ledger require to be considered when designing concurrency improvement mechanisms.

2) FUTURE WORKS

Smart contracts are expected to play a vital role in the industrial IoT context in future. The concurrency improvements must align with the restricted computing nature of IoT. The concurrency in consensus mechanisms must be efficient to fulfill the requirements of IoT.

V. OTHER FUTURE RESEARCH TOPICS

We next examine the possibly of applying other computer science theories to smart contracts in different aspects. The applicability and related works of significant theories of computer science to smart contracts discussed.

A. SMART CONTRACTS AND GAME THEORY

Game theory consists of a set of mathematical tools for identifying the interactions between agents. The combination of smart contracts and game theory highlights as emerging research topics. There are many applications that will be beneficial to smart contracts. Liu *et al.* [73] presented a comprehensive survey on the application of game theory to smart contracts. The authors discussed the applicability of game theory for different aspects of smart contracts such as security and mining management. The authors also highlighted the existing challenges and future research directions. Piasecki [151] discussed the game theory behind smart contract integrated casinos. The author explored that an attacker who is financially strong can game the system through purchasing computer power which is beneficial for himself. The author proposed how to secure the Proof-of-Work blockchain from the type of attack focused.

B. SMART CONTRACTS AND ARTIFICIAL INTELLIGENCE

Artificial Intelligence (AI) can be applied to the smart contracts in different ways. Some AI techniques can be embedded in the smart contract codes, others can be used to validate smart contracts. Furthermore, there are many emerging applications of Tensor and other deep learning concepts for the blockchain based smart contracts. Cognitive computing is another subset of AI which simulates human thoughts on computing infrastructure.

1) AI FOR TESTING AND EVALUATION OF SMART CONTRACTS

AI is generally applicable to the testing of smart contracts. More specifically, the testing can be focused on directions such as the performance testing, vulnerability detection, and correctness evaluation of the smart contracts. The contribution of AI as a utility service for the blockchain is important to improve the blockchain based smart contracts and the performance. Marwala *et al.* [152] presented how to use AI for the verification of smart contracts. The authors pointed out important applications of AI to the blockchain-based smart contract context, such as improvement of security, scalability, and so on. The authors also highlighted the applicability of AI-based formal verification for the evaluation of smart contracts.

2) FEDERATED LEARNING

Federated learning is a decentralized and collaborative learning approach which is aligned with the decentralization capability of the blockchain. Federated learning operates without uploading the raw data as the training datasets. Especially, the distributed sensitive information, such as healthcare information can be integrated with blockchain to achieve different functionalities, such as data access control and federated learning. The combination of federated learning and smart contracts can form new research opportunities. Lu *et al.* [153] proposed a blockchain and federated learning based privacy preservation mechanism for the industrial IoT. The authors integrated federated learning to the consensus to improve the computing resource consumption and efficiency in operation. The open issues associated with the resource-restricted computing infrastructure are also discussed to highlight the data privacy requirements. Kang *et al.* [154] proposed a federated learning system based on a consortium blockchain. The authors designed a contract theory-based incentive mechanism to evaluate high reputation workers for reliable training to elaborate on the learning process. The authors also discussed the improvement requirement of the reputation calculation.

3) SMART CONTRACTS AND COGNITIVE COMPUTING

Cognitive computing is an advanced AI research topic which enables human thinking in the computing infrastructure. Cognitive computing is adopted with the human thinking pattern and limitations in the execution which yields significantly higher accuracy when comparing with the other AI techniques. Blockchain-based smart contracts will improve the service values in the different application scenarios of cognitive computing. The data transparency, decentralized access control capability and the decentralized trust are the significant features of blockchain based smart contracts in the perspective of cognitive computing. Daniel *et al.* [155] conducted a survey on the applicability of cognitive computing in the healthcare domain. The authors highlighted that the implementation of blockchain for the healthcare is not a straightforward and it is important to meet the compliance requirements.

4) SMART CONTRACTS WITH TENSOR NETWORKS

There are significant applications in the tensor networks for smart contracts. These applications are relevant to industries such as financial and retail trade. The predictive modeling, the customer buying pattern analysis can be associated with blockchain and tensor networks. Charlie *et al.* [156] presented a tensor-based approach to predict smart contract interactions based on their cryptocurrency exchanges. The tensor modeling and stochastic process based approach utilized to underline the actual exchanges between smart contracts. The proposed approach is also capable of predicting future exchanges.

C. SMART CONTRACTS IN DATA SCIENCE

Smart contracts are applicable as a scalable technique in data science. Especially, controlling a massive data volume in a centralized architecture arises performance bottlenecks, failure risks and security risks. The role of smart contracts for data science is vital in different aspects. The applications of blockchain-based smart contracts for data science includes data access control, data integrity, ensuring decentralized trust, and enabling trusted data sharing mechanisms. Karafiloski [157] presented a survey on blockchain based solutions for big data. The authors reviewed different use cases such as medical record access control, IoT, and

digital property management. Abdullah *et al.* [158] detailed authentication techniques associated with blockchain for big data. The authors discussed the Kerberos authentication and how the blockchain is capable of addressing the limitations identified. Yue *et al.* [159] presents a data sharing platform which ensures traceability. The smart contracts can be used to enable the data sharing. Xu *et al.* [160] presented a smart contract-based storage system, named as Sapphire to support data analytics in IoT. Uchibeke *et al.* [161] developed a blockchain access control system using Hyperledger Fabric to control the access of large data sets.

VI. CONCLUSION

The paper starts with the concepts which are prerequisites for the blockchain-based smart contracts. The technical aspects of the smart contracts section provides a broad discussion on the essential features of smart contracts. The smart contracts and the current research of important topics in computer science also included the technical aspects section. The lessons learned and the future works section elaborated with an overview of the future research directions along with important insights from the technical aspects. As we can see, the application domains of smart contracts will expand the future. The gap between human and smart contracts will be eliminated in future through mobility. Smart contracts must be improved in the form of efficiency and transaction processing time and it will expose further opportunities to smart contracts. The next generation of the computing requires optimal and lightweight computation. Hence, the consensus mechanisms are required to improve to support the operation of blockchain on resource-constrained future computing infrastructure. The human-smart contract gap reduction will be a key research concern in future to improve the usability of smart contracts to solve the problems in the existing systems.

REFERENCES

[1] M. Garriga, S. D. Palma, M. Arias, A. Renzis, R. Pareschi, and D. A. Tamburri, "Blockchain and cryptocurrencies: A classification and comparison of architecture drivers," *Concurrency Comput., Pract. Exper.*, vol. 2, p. e5992, Oct. 2020.

[2] S. Cohen, A. Rosenthal, and A. Zohar, "Reasoning about the future in blockchain databases," in *Proc. IEEE 36th Int. Conf. Data Eng. (ICDE)*, Apr. 2020, pp. 1930–1933.

[3] G. Srivastava, R. M. Parizi, and A. Dehghantanha, "The future of blockchain technology in healthcare Internet of Things security," in *Blockchain Cybersecurity, Trust Privacy*. Springer, 2020, pp. 161–184.

[4] A. Bazarhanova, J. Magnusson, J. Lindman, E. Chou, and A. Nilsson, "Blockchain-based electronic identification: Cross-country comparison of six design choices," Tech. Rep., 2019.

[5] T. Aste, P. Tasca, and T. Di Matteo, "Blockchain technologies: The foreseeable impact on society and industry," *Computer*, vol. 50, no. 9, pp. 18–28, 2017.

[6] S. K. Lo, X. Xu, Y. K. Chiam, and Q. Lu, "Evaluating suitability of applying blockchain," in *Proc. 22nd Int. Conf. Eng. Complex Comput. Syst. (ICECCS)*, Nov. 2017, pp. 158–161.

[7] M. Pilkington, "Blockchain technology: Principles and applications," in *Research Handbook on Digital Transformations*. Broadheath, U.K.: Edward Elgar, 2016.

[8] M. J. M. Chowdhury, A. Colman, M. A. Kabir, J. Han, and P. Sarda, "Blockchain versus database: A critical analysis," in *Proc. 17th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun.*, Aug. 2018, pp. 1348–1353.

[9] F. Hofmann, S. Wurster, E. Ron, and M. Böhmecke-Schwafert, "The immutability concept of blockchains and benefits of early standardization," in *Proc. ITU Kaleidoscope, Challenges Data-Driven Soc. (ITU K)*, 2017, pp. 1–8.

[10] M. D. Pierro, "What is the blockchain?" *Comput. Sci. Eng.*, vol. 19, no. 5, pp. 92–95, 2017.

[11] A. Anjum, M. Sporny, and A. Sill, "Blockchain standards for compliance and trust," *IEEE Cloud Comput.*, vol. 4, no. 4, pp. 84–90, Jul. 2017.

[12] R. Zhang, R. Xue, and L. Liu, "Security and privacy on blockchain," *ACM Comput. Surv.*, vol. 52, pp. 1–34, Jul. 2019, doi: 10.1145/3316481.

[13] Q. Zhu, S. W. Loke, R. Trujillo-Rasua, F. Jiang, and Y. Xiang, "Applications of distributed ledger technologies to the Internet of Things: A survey," *ACM Comput. Surv.*, vol. 52, no. 6, pp. 1–34, Nov. 2019, doi: 10.1145/3359982.

[14] I. A. Qasse, M. Abu Talib, and Q. Nasir, "Inter blockchain communication: A survey," in *Proc. 6th Annu. Int. Conf. Res. Track*, New York, NY, USA, 2019, pp. 1–6, doi: 10.1145/3333165.3333167.

[15] W. Chen, Z. Xu, S. Shi, Y. Zhao, and J. Zhao, "A survey of blockchain applications in different domains," in *Proc. Int. Conf. Blockchain Technol. Appl.*, New York, NY, USA, 2018, pp. 17–21, doi: 10.1145/3301403.3301407.

[16] P. Chakraborty, R. Shahriyar, A. Iqbal, and A. Bosu, "Understanding the software development practices of blockchain projects: A survey," in *Proc. 12th ACM/IEEE Int. Symp. Empirical Softw. Eng. Meas.*, New York, NY, USA, Oct. 2018, p. 28, doi: 10.1145/3239235.3240298.

[17] P. T. Duy, D. T. T. Hien, D. H. Hien, and V.-H. Pham, "A survey on opportunities and challenges of blockchain technology adoption for revolutionary innovation," in *Proc. 9th Int. Symp. Inf. Commun. Technol.*, New York, NY, USA, 2018, pp. 200–207, doi: 10.1145/3287921.3287978.

[18] Btc. *Yes, Bitcoin Can Do Smart Contracts and Particl Demonstrates How*. [Online]. Available: https://bitcoinmagazine.com/articles/yes-bitcoin-can-do-smart-contracts-and-particl-demonstrates-how/

[19] S. Lande and R. Zunino, "SoK: Unraveling bitcoin smart contracts," *Princ. Secur.*, vol. 4, p. 217, Dec. 2018.

[20] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, Apr. 2014.

[21] Y. Hu, A. Manzoor, P. Ekparinya, M. Liyanage, K. Thilakarathna, G. Jourjon, A. Seneviratne, and M. E Ylianttila, "A delay-tolerant payment scheme based on the ethereum blockchain," 2018, *arXiv:1801.10295*. [Online]. Available: http://arxiv.org/abs/1801.10295

[22] D. Hopwood, S. Bowe, T. Hornby, and N. Wilcox, "Zcash protocol specification," Zerocoin Electric Coin Company, Boulder, CO, USA, Tech. Rep. 2016-1.10, 2016.

[23] E. Duffield and D. Diaz, "Dash: A privacy-centric cryptocurrency," Tech. Rep., 2015.

[24] M. T. Rosner and A. Kang, "Understanding and regulating twenty-first century payment systems: The ripple case study," *Michigan Law Rev.*, vol. 114, p. 649, Feb. 2015.

[25] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "MedRec: Using blockchain for medical data access and permission management," in *Proc. 2nd Int. Conf. Open Big Data (OBD)*, Aug 2016, pp. 25–30.

[26] P. Nichol and J. Brandt, "Co-creation of trust for healthcare: The cryptocitizen framework for interoperability with blockchain," Tech. Rep., Jul. 2016.

[27] T.-T. Kuo and L. Ohno-Machado, "ModelChain: Decentralized privacy-preserving healthcare predictive modeling framework on private blockchain networks," 2018, *arXiv:1802.01746*. [Online]. Available: http://arxiv.org/abs/1802.01746

[28] G. G. Dagher, J. Mohler, M. Milojkovic, and P. B. Marella, "Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology," *Sustain. Cities Soc.*, vol. 39, pp. 283–297, May 2018.

[29] X. Yue, H. Wang, D. Jin, M. Li, and W. Jiang, "Healthcare data gateways: Found healthcare intelligence on blockchain with novel privacy risk control," *J. Med. Syst.*, vol. 40, no. 10, p. 218, Oct. 2016.

[30] S. P. Novikov, O. D. Kazakov, N. A. Kulagina, and N. Y. Azarenko, "Blockchain and smart contracts in a decentralized health infrastructure," in *Proc. IEEE Int. Conf. Qual. Manage.*, Mar. 2018, pp. 697–703.

[31] S. Alexaki, G. Alexandris, V. Katos, and E. N. Petroulakis, "Blockchain-based electronic patient records for regulated circular healthcare jurisdictions," in *Proc. IEEE 23rd Int. Workshop Comput. Aided Model. Design Commun. Links Netw. (CAMAD)*, Mar. 2018, pp. 1–6.

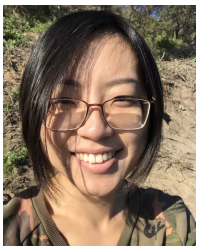
[32] T.-T. Kuo, H.-E. Kim, and L. Ohno-Machado, "Blockchain distributed ledger technologies for biomedical and health care applications," *J. Amer. Med. Inf. Assoc.*, vol. 24, no. 6, pp. 1211–1220, Nov. 2017.
[33] M. I. Mehar, C. L. Shier, A. Giambattista, E. Gong, G. Fletcher, R. Sanayhie, H. M. Kim, and M. Laskowski, "Understanding a revolutionary and flawed grand experiment in blockchain: The dao attack," *J. Cases Inf. Technol.*, vol. 21, no. 1, pp. 19–32, 2019.
[34] A. Wright and P. De Filippi, "Decentralized blockchain technology and the rise of lex cryptographia," Tech. Rep., 2015.
[35] K. Wust and A. Gervais, "Do you need a blockchain?" in *Proc. Crypto Valley Conf. Blockchain Technol. (CVCBT)*, Jun. 2018, pp. 45–54.
[36] C. D. Clack, V. A. Bakshi, and L. Braine, "Smart contract templates: Essential requirements and design options," 2016, *arXiv:1612.04496*. [Online]. Available: http://arxiv.org/abs/1612.04496
[37] S. Wang, Y. Yuan, X. Wang, J. Li, R. Qin, and F.-Y. Wang, "An overview of smart contract: Architecture, applications, and future trends," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 108–113.
[38] N. Reiff. (2020). *What Is ERC-20 and What Does It Mean for Ethereum*. [Online]. Available: https://www.investopedia.com/news/what-erc20-and-what-does-it-mean-ethereum/
[39] Z. Zheng, S. Xie, H.-N. Dai, W. Chen, X. Chen, J. Weng, and M. Imran, "An overview on smart contracts: Challenges, advances and platforms," *Future Gener. Comput. Syst.*, vol. 105, pp. 475–491, 2020.
[40] V. Buterin, "A next-generation smart contract and decentralized application platform," *White Paper*, vol. 3, p. 37, Jan. 2014.
[41] W. Corda. *Corda Technical Whitepaper*. [Online]. Available: https://www.corda.net/content/corda-technical-whitepaper.pdf
[42] H. Fabric. (2018). *Hyperledger Fabric*. [Online]. Available: https://www.hyperledger.org/wp-content/uploads/2018/07/
[43] R. Waves. *Waves*. https://wavesplatform.com/files/whitepaper_v0.pdf
[44] I. Bashir, *Mastering Blockchain: Distributed Ledger Technology, Decentralization, and Smart Contracts Explained*. Birmingham, U.K.: Packt, 2018.
[45] D. Macrinici, C. Cartofeanu, and S. Gao, "Smart contract applications within blockchain technology: A systematic mapping study," *Telematics Informat.*, vol. 35, no. 8, pp. 2337–2354, Dec. 2018.
[46] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *Proc. IEEE Int. Congr. Big Data (BigData congress)*, Mar. 2017, pp. 557–564.
[47] A. Norta, "Designing a smart-contract application layer for transacting decentralized autonomous organizations," in *Proc. Int. Conf. Adv. Comput. Data Sci.* Springer, 2016, pp. 595–604.
[48] T. Mancini-Griffoli, M. S. M. Peria, I. Agur, A. Ari, J. Kiff, A. Popescu, and C. Rochon, "Casting light on central bank digital currency," *IMF Staff Discussion Notes*, vol. 1, pp. 8–18, Dec. 2018.
[49] E. Shapiro, "Point: Foundations of E-democracy," *Commun. ACM*, vol. 61, no. 8, pp. 31–34, 2018.
[50] S. S. Gupta, *Blockchain*. Hoboken, NJ, USA: Wiley, 2017.
[51] F. Rizal Batubara, J. Ubacht, and M. Janssen, "Unraveling transparency and accountability in blockchain," in *Proc. 20th Annu. Int. Conf. Digit. Government Res.*, Jun. 2019, pp. 204–213.
[52] T. Nugent, D. Upton, and M. Cimpoesu, "Improving data transparency in clinical trials using blockchain smart contracts," *FResearch*, vol. 5, p. 2541, Oct. 2016.
[53] F. Yiannas, "A new era of food transparency powered by blockchain," *Innovations: Technol., Governance, Globalization*, vol. 12, nos. 1–2, pp. 46–56, Jul. 2018.
[54] M. Farnaghi and A. Mansourian, "Blockchain, an enabling technology for transparent and accountable decentralized public participatory gis," *Cities*, vol. 105, Mar. 2020, Art. no. 102850.
[55] N. N. Pokrovskaia, "Tax, financial and social regulatory mechanisms within the knowledge-driven economy. Blockchain algorithms and fog computing for the efficient regulation," in *Proc. IEEE Int. Conf. Soft Comput. Meas. (SCM)*, May 2017, pp. 709–712.
[56] F. Sander, J. Semeijn, and D. Mahr, "The acceptance of blockchain technology in meat traceability and transparency," *Brit. Food J.*, vol. 120, no. 9, pp. 2066–2079, Sep. 2018.
[57] C. Cachin, "Architecture of the hyperledger blockchain fabric," in *Proc. Workshop Distrib. Cryptocurrencies Consensus Ledgers*, 2016, vol. 310, no. 4, pp. 1–3.
[58] C. Dannen, *Introducing Ethereum Solidity*, vol. 1. Springer, 2017.
[59] G. O. Karame, E. Androulaki, M. Roeschlin, A. Gervais, and S. Æapkun, "Misbehavior in bitcoin: A study of double-spending and accountability," *ACM Trans. Inf. Syst. Secur.*, vol. 18, no. 1, pp. 1–32, Jun. 2015.
[60] R. Nem. *NEM Technical Reference*. [Online]. Available: https://nem.io/wp-content/themes/nem/files/NEM_techRef.pdf/
[61] W. Stellar. *Stellar Whitepaper*. [Online]. Available: https://stellargold.net/whitepaper.pdf
[62] E. Classic. *Ethereum Classic*. Accessed: Oct. , 2017. [Online]. Available: https://ethereumclassic.github.io
[63] L. Goodman. (2014). *Tezos—A Self-Amending Crypto-Ledger White Paper*. [Online]. Available: https://www.tezos.com/static/papers/white_paper.pdf
[64] *Neo Blockchain Whitepaper*. [Online]. Available: https://docs.neo.org/docs/en-us/index.html
[65] *Cardano Blockchain Whitepaper*. [Online]. Available: https://www.circle.com/marketing/pdfs/research/circle-research-cardano.pdf
[66] *EOS Blockchain Whitepaper*. [Online]. Available: https://github.com/BlockchainTranslator/EOS/blob/master/TechDoc/EOS.IO-Technical-WhitePaper-v2.md
[67] *Qtum Blockchain Whitepaper*. [Online]. Available: https://old.qtum.org/user/pages/03.tech/01.white-papers/QtumWhitepaper.pdf
[68] *Lisk Whitepaper*. [Online]. Available: https://github.com/slasheks/lisk-whitepaper/blob/development/LiskWhitepaper.md
[69] *Ark White Paper*.
[70] *Rootstock Platform Bitcoin Powered Smart Contract Whitepaper*. [Online]. Available: https://www.rsk.co/wp-content/uploads/2019/02/RSK-White-Paper-Updated.pdf
[71] *NXT Whitepaper*. [Online]. Available: https://www.rsk.co/wp-content/uploads/2019/02/RSK-White-Paper-Updated.pdf
[72] T. Min and W. Cai, "A security case study for blockchain games," in *Proc. IEEE Games, Entertainment, Media Conf. (GEM)*, Jun. 2019, pp. 1–8.
[73] Z. Liu, N. C. Luong, W. Wang, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "A survey on blockchain: A game theoretical perspective," *IEEE Access*, vol. 7, pp. 47615–47643, 2019.
[74] Y. Zhang, S. Ma, J. Li, K. Li, S. Nepal, and D. Gu, "SMARTSHIELD: Automatic smart contract protection made easy," in *Proc. IEEE 27th Int. Conf. Softw. Anal., Evol. Reeng. (SANER)*, Feb. 2020, pp. 23–34.
[75] C. G. Harris, "The risks and challenges of implementing ethereum smart contracts," in *Proc. IEEE Int. Conf. Blockchain Cryptocurrency (ICBC)*, May 2019, pp. 104–107.
[76] S. Kim and S. Ryu, "Analysis of blockchain smart contracts: Techniques and insights," in *Proc. IEEE Secure Develop. (SecDev)*. 2020, pp. 65–73.
[77] P. Otte, M. de Vos, and J. Pouwelse, "TrustChain: A sybil-resistant scalable blockchain," *Future Gener. Comput. Syst.*, vol. 107, pp. 770–780, Jun. 2020.
[78] Y. Cai and D. Zhu, "Fraud detections for online businesses: A perspective from blockchain technology," *Financial Innov.*, vol. 2, no. 1, p. 20, Dec. 2016.
[79] M. Bellare, Z. Brakerski, M. Naor, T. Ristenpart, G. Segev, H. Shacham, and S. Yilek, "Hedged public-key encryption: How to protect against bad randomness," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Springer, 2009, pp. 232–249.
[80] K. Chatterjee, A. K. Goharshady, and A. Pourdamghani, "Probabilistic smart contracts: Secure randomness on the blockchain," in *Proc. IEEE Int. Conf. Blockchain Cryptocurrency (ICBC)*, May 2019, pp. 403–412.
[81] D. Efanov and P. Roschin, "The all-pervasiveness of the blockchain technology," *Procedia Comput. Sci.*, vol. 123, pp. 116–121, 2018.
[82] U. W. Chohan, "The double spending problem and cryptocurrencies," *SSRN Electron. J.*, vol. 2, pp. 1–6, Dec. 2017.
[83] S. Zhang and J.-H. Lee, "Double-spending with a sybil attack in the bitcoin decentralized network," *IEEE Trans. Ind. Informat.*, vol. 15, no. 10, pp. 5715–5722, Mar. 2019.
[84] S. Dey, "Securing majority-attack in blockchain using machine learning and algorithmic game theory: A proof of work," in *Proc. 10th Comput. Sci. Electron. Eng. (CEEC)*, 2018, pp. 7–10.
[85] J. Moubarak, E. Filiol, and M. Chamoun, "On blockchain security and relevant attacks," in *Proc. IEEE Middle East North Afr. Commun. Conf. (MENACOMM)*, Mar. 2018, pp. 1–6.
[86] A. Groce, J. Feist, G. Grieco, and M. Colburn, "What are the actual flaws in important smart contracts (and how can we find them)?" in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Springer, 2020, pp. 634–653.
[87] C. Liu, J. Gao, Y. Li, H. Wang, and Z. Chen, "Studying gas exceptions in blockchain-based cloud applications," *J. Cloud Comput.*, vol. 9, no. 1, pp. 1–25, Dec. 2020.
[88] H. Hasanova, U.-J. Baek, M.-G. Shin, K. Cho, and M.-S. Kim, "A survey on blockchain cybersecurity vulnerabilities and possible countermeasures," *Int. J. Netw. Manage.*, vol. 29, no. 2, p. e2060, Mar. 2019.

[89] X. Li, P. Jiang, T. Chen, X. Luo, and Q. Wen, "A survey on the security of blockchain systems," *Future Gener. Comput. Syst.*, vol. 107, pp. 841–853, Jun. 2020.
[90] A. Singh, R. M. Parizi, Q. Zhang, K.-K.-R. Choo, and A. Dehghantanha, "Blockchain smart contracts formalization: Approaches and challenges to address vulnerabilities," *Comput. Secur.*, vol. 88, Jan. 2020, Art. no. 101654.
[91] D. K. Tosh, S. Shetty, X. Liang, C. A. Kamhoua, K. A. Kwiat, and L. Njilla, "Security implications of blockchain cloud with analysis of block withholding attack," in *Proc. 17th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput. (CCGRID)*, May 2017, pp. 458–467.
[92] S. Tikhomirov, E. Voskresenskaya, I. Ivanitskiy, R. Takhaviev, E. Marchenko, and Y. and Alexandrov, "Smartcheck: Static Analysis of Ethereum Smart Contracts," in *Proc. IEEE/ACM 1st Int. Workshop Emerg. Trends Softw. Eng. Blockchain (WETSEB)*, 2018, pp. 9–16.
[93] I. Nikoliá, A. Kolluri, I. Sergey, P. Saxena, and A. Hobor, "Finding the greedy, prodigal, and suicidal contracts at scale," in *Proc. 34th Annu. Comput. Secur. Appl. Conf.*, Dec. 2018, pp. 653–663.
[94] C. Liu, H. Liu, Z. Cao, Z. Chen, B. Chen, and B. Roscoe, "ReGuard: Finding reentrancy bugs in smart contracts," in *Proc. 40th Int. Conf. Softw. Eng.*, May 2018, pp. 65–68.
[95] B. Jiang, Y. Liu, and W. K. Chan, "ContractFuzzer: Fuzzing smart contracts for vulnerability detection," in *Proc. 33rd ACM/IEEE Int. Conf. Automated Softw. Eng.*, Sep. 2018, pp. 259–269.
[96] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 254–269.
[97] H. Liu, C. Liu, W. Zhao, Y. Jiang, and J. Sun, "S-gram: Towards semantic-aware security auditing for ethereum smart contracts," in *Proc. 33rd ACM/IEEE Int. Conf. Automated Softw. Eng.*, Sep. 2018, pp. 814–819.
[98] L. Brent, A. Jurisevic, M. Kong, E. Liu, F. Gauthier, V. Gramoli, R. Holz, and B. Scholz, "Vandal: A scalable security analysis framework for smart contracts," 2018, *arXiv:1809.03981*. [Online]. Available: http://arxiv.org/abs/1809.03981
[99] M. Suiche, "Porosity: A decompiler for blockchain-based smart contracts bytecode," *Def-Con*, vol. 25, p. 11, Jul. 2017.
[100] N. Grech, M. Kong, A. Jurisevic, L. Brent, B. Scholz, and Y. Smaragdakis, "Madmax: Surviving out-of-gas conditions in ethereum smart contracts," *Proc. ACM Program. Lang.*, vol. 2, p. 116, Mar. 2018.
[101] K. Wüst and A. Gervais, "Ethereum Eclipse Attacks," ETH Zürich, Zürich, Switzerland, Tech. Rep., 2016.
[102] P. Tsankov, A. Dan, D. Drachsler-Cohen, A. Gervais, F. Bánzli, and M. Vechev, "Securify: Practical security analysis of smart contracts," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2018, pp. 67–82.
[103] I. Grishchenko, M. Maffei, and C. Schneidewind, "A semantic framework for the security analysis of ethereum smart contracts," in *Proc. Int. Conf. Princ. Secur. Trust*. Springer, 2018, pp. 243–269.
[104] S. Kalra, S. Goel, M. Dhawan, and S. Sharma, "ZEUS: Analyzing safety of smart contracts," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2018, pp. 18–21.
[105] I. Grishchenko, M. Maffei, and C. Schneidewind, "EtherTrust: Sound static analysis of ethereum bytecode," Technische Univ. Wien, Vienna, Austria, Tech. Rep., 2018.
[106] M. Mossberg, F. Manzano, E. Hennenfent, A. Groce, G. Grieco, J. Feist, T. Brunson, and A. Dinaburg, "Manticore: A user-friendly symbolic execution framework for binaries and smart contracts," 2019, *arXiv:1907.03890*. [Online]. Available: http://arxiv.org/abs/1907.03890
[107] P. Mell, J. Kelsey, and J. Shook, "Cryptocurrency smart contracts for distributed consensus of public randomness," in *Proc. Int. Symp. Stabilization, Saf., Secur. Distrib. Syst.* Springer, 2017, pp. 410–425.
[108] G. Destefanis, M. Marchesi, M. Ortu, R. Tonelli, A. Bracciali, and R. Hierons, "Smart contracts vulnerabilities: A call for blockchain software engineering?" in *Proc. Int. Workshop Blockchain Oriented Softw. Eng. (IWBOSE)*, 2018, pp. 19–25.
[109] N. Atzei, M. Bartoletti, and T. Cimoli, "A survey of attacks on ethereum smart contracts (SOK)," in *Proc. Int. Conf. Princ. Secur. Trust*. Springer, 2017, pp. 164–186.
[110] K. Delmolino, M. Arnett, A. Kosba, A. Miller, and E. Shi, "Step by step towards creating a safe smart contract: Lessons and insights from a cryptocurrency lab," in *Int. Conf. Financial Cryptogr. Data Secur.* Springer, 2016, pp. 79–94.
[111] M. Wohrer and U. Zdun, "Smart contracts: Security patterns in the Ethereum ecosystem and solidity," in *Proc. Int. Workshop Blockchain Oriented Softw. Eng. (IWBOSE)*, Mar. 2018, pp. 2–8.
[112] R. M. Parizi, A. Dehghantanha, K.-K. R. Choo, and A. Singh, "Empirical vulnerability analysis of automated smart contracts security testing on blockchains," in *Proc. 28th Annu. Int. Conf. Comput. Sci. Softw. Eng.*, 2018, pp. 103–113.
[113] A. Mavridou and A. Laszka, "Designing secure ethereum smart contracts: A finite state machine based approach," 2017, *arXiv:1711.09327*. [Online]. Available: http://arxiv.org/abs/1711.09327
[114] K. Bhargavan, A. Delignat-Lavaud, C. Fournet, A. Gollamudi, G. Gonthier, N. Kobeissi, A. Rastogi, T. Sibut-Pinote, N. Swamy, and S. Zanella-Béguelin, "Short paper: Formal verification of smart contracts," in *Proc. 11th ACM Workshop Program. Lang. Anal. Secur. (PLAS)*, 2016, pp. 91–96.
[115] T. Abdellatif and K.-L. Brousmiche, "Formal verification of smart contracts based on users and blockchain behaviors models," in *Proc. 9th IFIP Int. Conf. New Technol., Mobility Secur. (NTMS)*, Feb. 2018, pp. 1–5.
[116] Z. Nehai, P.-Y. Piriou, and F. Daumas, "Model-checking of smart contracts," in *Proc. IEEE Int. Conf. Internet Things*, Jul. 2018, pp. 980–987.
[117] E. Albert, P. Gordillo, B. Livshits, A. Rubio, and I. Sergey, "EthIR: A framework for high-level analysis of ethereum bytecode," in *Proc. Int. Symp. Automated Technol. Verification Anal.* Springer, 2018, pp. 513–520.
[118] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2016, pp. 839–858.
[119] S. R. Niya, F. Shupfer, T. Bocek, and B. Stiller, "Setting up flexible and light weight trading with enhanced user privacy using smart contracts," in *Proc. IEEE/IFIP Netw. Oper. Manage. Symp.*, Apr. 2018, pp. 1–2.
[120] D. Chatzopoulos, S. Gujar, B. Faltings, and P. Hui, "Privacy preserving and cost optimal mobile crowdsensing using smart contracts on blockchain," in *Proc. IEEE 15th Int. Conf. Mobile Ad Hoc Sensor Syst. (MASS)*, Mar. 2018, pp. 442–450.
[121] X. Liang, S. Shetty, D. Tosh, C. Kamhoua, K. Kwiat, and L. Njilla, "ProvChain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability," in *Proc. 17th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput. (CCGRID)*, May 2017, pp. 468–477.
[122] M. Al-Bassam, A. Sonnino, S. Bano, D. Hrycyszyn, and G. Danezis, "Chainspace: A sharded smart contracts platform," 2017, *arXiv:1708.03778*. [Online]. Available: http://arxiv.org/abs/1708.03778
[123] H. Kalodner, S. Goldfeder, X. Chen, S. M. Weinberg, and E. W. Felten, "Arbitrum: Scalable, private smart contracts," in *Proc. 27th Secur. Symp.*, 2018, pp. 1353–1370.
[124] R. Cheng, F. Zhang, J. Kos, W. He, N. Hynes, N. Johnson, A. Juels, A. Miller, and D. Song, "Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Jun. 2019, pp. 185–200.
[125] F. Zhang, E. Cecchetti, K. Croman, A. Juels, and E. Shi, "Town crier: An authenticated data feed for smart contracts," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 270–282.
[126] R. Yuan, Y.-B. Xia, H.-B. Chen, B.-Y. Zang, and J. Xie, "Shadoweth: Private smart contract on public blockchain," *J. Comput. Sci. Technol.*, vol. 33, no. 3, pp. 542–556, 2018.
[127] F. Benhamouda, S. Halevi, and T. Halevi, "Supporting private data on hyperledger fabric with secure multiparty computation," *IBM J. Res. Develop.*, vol. 63, no. 2/3, pp. 1–8, Mar. 2019.
[128] G. Zyskind, O. Nathan, and A. Pentland, "Enigma: Decentralized computation platform with guaranteed privacy," 2015, *arXiv:1506.03471*. [Online]. Available: http://arxiv.org/abs/1506.03471
[129] B. Bünz, S. Agrawal, M. Zamani, and D. Boneh, "Zether: Towards privacy in a smart contract world," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Springer, 2020, pp. 423–443.
[130] H. Halpin and M. Piekarska, "Introduction to security and privacy on the blockchain," in *Proc. IEEE Eur. Symp. Secur. Privacy Workshops*, Apr. 2017, pp. 1–3.
[131] Z. Liehuang, G. Feng, S. Meng, L. Yandong, Z. Baokun, M. Hongliang, and W. Zhen, "Survey on privacy preserving techniques for blockchain technology," *J. Comput. Res. Develop.*, p. 2170, 2017.
[132] B. K. Mohanta, D. Jena, S. S. Panda, and S. Sobhanayak, "Blockchain technology: A survey on applications and security privacy challenges," *Internet Things*, vol. 8, Dec. 2019, Art. no. 100107.

[133] R. Henry, A. Herzberg, and A. Kate, "Blockchain access privacy: Challenges and directions," *IEEE Secur. Privacy*, vol. 16, no. 4, pp. 38–45, Jul. 2018.
[134] Q. Feng, D. He, S. Zeadally, M. K. Khan, and N. Kumar, "A survey on privacy protection in blockchain system," *J. Netw. Comput. Appl.*, vol. 126, pp. 45–58, Jan. 2019.
[135] R. Gupta, S. Tanwar, F. Al-Turjman, P. Italiya, A. Nauman, and S. W. Kim, "Smart contract privacy protection using ai in cyber-physical systems: Tools, techniques and challenges," *IEEE Access*, vol. 8, pp. 24746–24772, 2020.
[136] N. Kapsoulis, A. Psychas, G. Palaiokrassas, A. Marinakis, A. Litke, and T. Varvarigou, "Know your customer (KYC) implementation with smart contracts on a privacy-oriented decentralized architecture," *Future Internet*, vol. 12, no. 2, p. 41, Feb. 2020.
[137] A. Dorri, M. Steger, S. S. Kanhere, and R. Jurdak, "BlockChain: A distributed solution to automotive security and privacy," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 119–125, Dec. 2017.
[138] S. Woo, J. Song, and S. Park, "A distributed oracle using intel sgx for blockchain-based iot applications," *Sensors*, vol. 20, no. 9, p. 2725, 2020.
[139] G. Ayoade, V. Karande, L. Khan, and K. Hamlen, "Decentralized IoT data management using BlockChain and trusted execution environment," in *Proc. IEEE Int. Conf. Inf. Reuse Integr. (IRI)*, Jul. 2018, pp. 15–22.
[140] S. Felsen, Á. Kiss, T. Schneider, and C. Weinert, "Secure and private function evaluation with intel SGX," in *Proc. ACM SIGSAC Conf. Cloud Comput. Secur. Workshop*, 2019, pp. 165–181.
[141] Z. Bao, Q. Wang, W. Shi, L. Wang, H. Lei, and B. Chen, "When blockchain meets SGX: An overview, challenges, and open issues," *IEEE Access*, vol. 8, pp. 170404–170420, 2020.
[142] L.-D. Ibá nez, K. O'Hara, and E. Simperl, "On blockchains and the general data protection regulation," Tech. Rep., 2018.
[143] A. Juels, A. Kosba, and E. Shi, "The ring of gyges: Investigating the future of criminal smart contracts," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 283–295.
[144] Intel. (2020). *Intel Software Guard Extensions (Intel SGX)*. [Online]. Available: https://www.intel.com.au/content/www/au/en/architecture-and-technology/software-guard-extensions.html
[145] WikiPedia. (2020). *Yao's Millionaires' Problem*. [Online]. Available: https://splash247.com/blockchain-roaming-in-the-maritime-industry/
[146] F. Idelberger, G. Governatori, R. Riveret, and G. Sartor, "Evaluation of logic-based smart contracts for blockchain systems," in *Proc. Int. Symp. Rules Rule Markup Lang. Semantic Web*. Springer, 2016, pp. 167–183.
[147] T. Chen, X. Li, X. Luo, and X. Zhang, "Under-optimized smart contracts devour your money," in *Proc. IEEE 24th Int. Conf. Softw. Anal., Evol. Reeng. (SANER)*, Feb. 2017, pp. 442–446.
[148] A. Kothapalli, A. Miller, and N. Borisov, "Smartcast: An incentive compatible consensus protocol using smart contracts," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Springer, 2017, pp. 536–552.
[149] A. Zhang and K. Zhang, "Enabling concurrency on smart contracts using multiversion ordering," in *Proc. Joint Int. Conf. Web Big Data*. Springer, 2018, pp. 425–439.
[150] P. S. Anjana, S. Kumari, S. Peri, S. Rathor, and A. Somani, "An efficient framework for optimistic concurrent execution of smart contracts," in *Proc. 27th Euromicro Int. Conf. Parallel, Distrib. Netw.-Based Process. (PDP)*, Feb. 2019, pp. 83–92.
[151] P. J. Piasecki, "Gaming self-contained provably fair smart contract casinos," *Ledger*, vol. 1, pp. 99–110, Dec. 2016.
[152] T. Marwala and B. Xing, "Blockchain and artificial intelligence," 2018, *arXiv:1802.04451*. [Online]. Available: http://arxiv.org/abs/1802.04451
[153] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial iot," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4177–4186, Jun. 2019.
[154] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10700–10714, Dec. 2019.
[155] J. Daniel, U. Lockheed MartinDallasTX, A. Sargolzaei, M. Abdelghani, S. Sargolzaei, and B. Amaba, "Blockchain technology, cognitive computing, and healthcare innovations," *J. Adv. Inf. Technol.*, pp. 194–198, 2017.
[156] J. Charlier, "Profiling smart contracts interactions with tensor decomposition and graph mining," Tech. Rep., 2017.
[157] E. Karafiloski and A. Mishev, "Blockchain solutions for big data challenges: A literature review," in *Proc. 17th Int. Conf. Smart Technol.*, Jul. 2017, pp. 763–768.
[158] N. Abdullah, A. Hakansson, and E. Moradian, "Blockchain based approach to enhance big data authentication in distributed environment," in *Proc. 9th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Jul. 2017, pp. 887–892.
[159] L. Yue, H. Junqin, Q. Shengzhi, and W. Ruijin, "Big data model of security sharing based on blockchain," in *Proc. 3rd Int. Conf. Big Data Comput. Commun. (BIGCOM)*, Aug. 2017, pp. 117–121.
[160] Q. Xu, K. M. M. Aung, Y. Zhu, and K. L. Yong, "A blockchain-based storage system for data analytics in the Internet of Things," in *New Advaces Internet Things*. Springer, 2018, pp. 119–138.
[161] U. Ugobame Uchibeke, K. A. Schneider, S. Hosseinzadeh Kassani, and R. Deters, "Blockchain access control ecosystem for big data security," in *Proc. IEEE Int. Conf. Internet Things*, Jul. 2018, pp. 1373–1378.

THARAKA MAWANANE HEWA (Student Member, IEEE) received the bachelor's degree in computer science and the M.Sc. degree (Hons.) in information security from the School of Computing, University of Colombo, Sri Lanka, in 2013 and 2016, respectively. He is currently pursuing the Ph.D. degree with the Centre for Wireless Communication, University of Oulu, Finland.

From 2012 to 2017, he worked in a leading digital payment solution provider in Sri Lanka, as a Senior Software Engineer. Within his career in the industry, he contributed to many projects, such as mobile banking, internet banking, PKI, and Automated Teller Machines, and involved in the system integration and support. He is a Certified Engineer with SafeNet Luna SA 6.0 HSM. In 2017, he joined Nanyang Technological University as a Research Associate. He played a vital role in many research and implementation projects in different contexts. He contributed to cybersecurity, digital payment systems, ongoing research, and implementation projects, including blockchain for 3D printing, agriculture, luxury watches, music asset monetization, and aviation. In 2019, he joined the Centre for Wireless Communication and his research focus in developing a blockchain-based platform as a service for industrial IoT. He has co-authored two publications related to the blockchain applications in the industry with contributing to one patent. His research interests include blockchain, PKI, 5G, banking systems security, healthcare security, and smart cities.



YINING HU (Student Member, IEEE) received the B.E. degree (Hons.) in electrical engineering from The University of Sydney, Sydney, Australia, and the Harbin Institute of Technology, Harbin, China, in 2016 (joint program), and the Ph.D. degree from the University of New South Wales, Sydney, in 2020. During her Ph.D. studies, she was also affiliated with Data61-CSIRO, Sydney. She joined IBM Australia, Melbourne, VIC, Australia, as a Postdoctoral Researcher, in 2020. Her main areas of research interests include illicit cryptocurrency transaction analysis, improving the utility of blockchain platforms, and enabling blockchain interoperability.

MADHUSANKA LIYANAGE (Senior Member, IEEE) received the B.Sc. degree (Hons.) in electronics and telecommunication engineering from the University of Moratuwa, Moratuwa, Sri Lanka, in 2009, the M.Eng. degree from the Asian Institute of Technology, Bangkok, Thailand, in 2011, the M.Sc. degree from the University of Nice Sophia Antipolis, Nice, France, in 2011, and the Ph.D. degree in communication engineering from the University of Oulu, Oulu, Finland, in 2016.

From 2011 to 2012, he worked a Research Scientist with the I3S Laboratory and Inria, Sophia Antipolis, France. He has been a Visiting Research Fellow with the Department of Computer Science, University of Oxford, Data61, CSIRO, Sydney, Australia, the Infolabs21, Lancaster University, U.K., and the School of Computer Science and Engineering, The University of New South Wales, from 2015 to 2018. He is currently an Ad Astra Fellow/an Assistant Professor with the School of Computer Science, University College Dublin, Ireland. He is also an Adjunct Professor with the University of Oulu, Finland. He has worked for more than 12 EU and international and national projects in ICT domain. He held responsibilities as a leader of work packages in several national and EU projects. He is currently the Finnish National Coordinator for EU COST Action CA15127 on resilient communication services. He is/was also serving as a Management Committee Member for four other EU COST action projects, namely EU COST Action IC1301, IC1303, CA15107, and CA16226. He has over three years' experience in research project management, research group leadership, research project proposal preparation, project progress documentation, and graduate student co-supervision/mentoring skills. He has been awarded three research grants and 22 other prestigious awards/scholarships during his research career. He has co-authored over 90 publications including two edited books with Wiley and one patent. His research interests include 5G, SDN, IoT, Blockchain, MEC, mobile, and virtual network security.

Dr. Liyanage served as a Technical Program Committee Members for EAI M3Apps 2016, 5GU 2017, EUCNC 2017, EUCNC 2018, 5GWF 2018, MASS 2018, MCWN 2018, WCNC 2019, EUCNC 2019, EUCNC 2020, MASS 2020, and ICBC 2021 conferences, and a Technical Program Co-Chair for SecureEdge workshop at IEEE CIT2017 conference and Blockchain for IoT workshop at IEEE Globecom 2018, IEEE ICC 2020, and IEEE 5GWF 2020. He has received two best paper awards in the areas of SDMN security (from NGMAST 2015) and 5G Security (from IEEE CSCN 2017). In 2015, 2016, and 2017, he won the Best Researcher Award from the Centre for Wireless Communications, University of Oulu, for his excellent contribution in project management and dissemination activities, and two of the research projects (MEVICO and SIGMONA projects) received the CELTIC Excellence Award, in 2013 and 2017, respectively, the prestigious Marie Skłodowska-Curie Actions Individual Fellowship from 2018 to 2020, and the 2020 IEEE ComSoc Outstanding Young Researcher Award from IEEE ComSoc EMEA, in 2020. He has served as a Session Chair for a number of other conferences, including IEEE WCNC 2013, CROWNCOM 2014, 5GU 2014, IEEE CIT 2017, IEEE PIMRC 2017, 5GWF 2018, Bobynet 2018, Globecom 2018, WCNC 2019, and ICC 2020. He is a Demo Co-Chair of WCNC 2019, the Publicity Chair of ISWCS 2019, and the Poster Chair of 6G Summit 2020. http://madhusanka.com.

SALIL S. KANHARE (Senior Member, IEEE) received the M.S. and Ph.D. degrees in electrical engineering from Drexel University, Philadelphia. He has held visiting appointments with the Institute of Infocom Research Singapore, Technical University Darmstadt, the University of Zurich, and the Graz University of Technology. He is currently a Professor with the School of Computer Science and Engineering, UNSW Sydney, Australia. He also holds affiliations with CSIRO's Data61 and the Cybersecurity Cooperative Research Centre. He has published over 250 peer-reviewed articles and delivered over 50 tutorials and keynote talks on these research topics. His research has been featured on ABC News Australia, Forbes, Wired, ZDNET, MIT Technology Review, Computer World, IEEE Spectrum, and other media outlets. His research interests include the Internet of Things, pervasive computing, cyber-physical systems, blockchain, cybersecurity, and applied machine learning. He is a Senior Member of the ACM. He regularly features on the organizing committee of a number of IEEE and ACM international conferences. He also serves on the Executive Committee for the IEEE Computer Society's Technical Committee on Computer Communications (TCCC). He was a recipient of the Alexander von Humboldt Research Fellowship. He has received eight Best Paper Awards. He is the General Chair of the IEEE International Conference on Blockchain and Cryptocurrency (IEEE ICBC) in 2021. He is the Editor-in-Chief of the *Ad Hoc Networks* Journal and on the Editorial Board of IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, *Pervasive and Mobile Computing* and *Computer Communications*.

MIKA YLIANTTILA (Senior Member, IEEE) received the Ph.D. degree in communications engineering from the University of Oulu, Finland, in 2005. He was the Director of the Center for Internet Excellence, from 2012 to 2015, a Vice Director of the Mediateam Oulu Research Group, from 2009 to 2011, and a Professor (Pro Tem) in computer science and engineering, and the Director of Information Networks Study Programme, from 2005 to 2010. He is currently a full-time Associate Professor (Tenure Track) with the Centre for Wireless Communications (CWC), Faculty of Information Technology and Electrical Engineering (ITEE), University of Oulu. He is also leading a research team and the Director of the Communications Engineering Doctoral Degree Program. He has coauthored more than 150 international peer-reviewed articles. His research interests include edge computing, network security, network virtualization, and software-defined networking. He is an Editor of *Wireless Networks* journal.