A Study on Blockchain Architecture Design Decisions and Their Security Attacks and Threats

SABREEN AHMADJEE, University of Birmingham, UK, Umm Al-Qura University, Saudi Arabia
CARLOS MERA-GÓMEZ, ESPOL Polytechnic University, Escuela Superior Politécnica del Litoral, ESPOL, Ecuador
RAMI BAHSOON, University of Birmingham, UK
RICK KAZMAN, University of Hawaii, USA

Blockchain is a disruptive technology intended to implement secure decentralised distributed systems, in which transactional data can be shared, stored, and verified by participants of the system without needing a central authentication/verification authority. Blockchain-based systems have several architectural components and variants, which architects can leverage to build secure software systems. However, there is a lack of studies to assist architects in making architecture design and configuration decisions for blockchain-based systems. This knowledge gap may increase the chance of making unsuitable design decisions and producing configurations prone to potential security risks. To address this limitation, we report our comprehensive systematic literature review to derive a taxonomy of commonly used architecture design decisions in blockchain-based systems. We map each of these decisions to potential security attacks and their posed threats. MITRE's attack tactic categories and Microsoft STRIDE threat modeling are used to systematically classify threats and their associated attacks to identify potential attacks and threats in blockchain-based systems. Our mapping approach aims to guide architects to make justifiable design decisions that will result in more secure implementations.

CCS Concepts: • **Security and privacy** → *Software and application security*; **Software security engineering**;

Additional Key Words and Phrases: Blockchain, security threat classification, architecture decision, design decisions

ACM Reference format:
Sabreen Ahmadjee, Carlos Mera-Gómez, Rami Bahsoon, and Rick Kazman. 2022. A Study on Blockchain Architecture Design Decisions and Their Security Attacks and Threats. *ACM Trans. Softw. Eng. Methodol.* 31, 2, Article 36e (March 2022), 45 pages.
https://doi.org/10.1145/3502740

Authors' addresses: S. Ahmadjee, University of Birmingham, School of Computer Science, Edgbaston, Birmingham, B15 2TT, UK, and Umm Al-Qura University, College of Computer and Information Systems, Makkah, Saudi Arabia; email: smahmadjee@uqu.edu.sa; C. Mera-Gómez, ESPOL Polytechnic University, Escuela Superior Politécnica del Litoral, ESPOL, Facultad de Ingeniería en Electricidad y Computación, Campus Gustavo Galindo Km 30.5 Vía Perimetral, P.O. Box 09-01-5863, Guayaquil, Ecuador; email: cjmera@espol.edu.ec; R. Bahsoon, University of Birmingham, School of Computer Science, Edgbaston, Birmingham, B15 2TT, UK; email: r.bahsoon@cs.bham.ac.uk; R. Kazman, University of Hawaii, Information Technology Management, 2500 Campus Rd, Honolulu, Hawaii, HI 96822; email: kazman@hawaii.edu.

1049-331X/2022/03-ART36e $15.00
https://doi.org/10.1145/3502740

1 INTRODUCTION

Blockchain technology has received widespread attention in both industry and academia since the success of Bitcoin, a seminal application based on this technology [169]. The use of blockchain technology has gone beyond cryptocurrency systems to underlie many dependable mainstream software systems including finance, integrity verification, the **Internet of Things (IoT)**, healthcare, data management, security, and privacy [25]. Organisations are already leveraging blockchain as a critical component within software system architectures to provide more dependable and secure computation and storage [25, 147]. Examples of organisations using blockchain include Health Linkages[1] and Hashed Health.[2] Blockchain is being used in many different application domains [179], for instance, supply chains energy sector, and insurance. In Reference [25], the authors represented and analysed blockchain-enabled applications across 21 distinct sectors. Further details about the taxonomy and classification of blockchain-based applications can be found in References [179] and [25].

Blockchain is a decentralised technology that claims to provide several security properties, including immutability, integrity, non-repudiation, and availability [169]. Indeed, blockchain is a chain of ordered blocks, which are distributed across thousands of nodes, each block connecting to the previous block via a cryptographic hash of its content. Thus, the block is seen as immutable in practice, since it cannot be modified retroactively without the modification of all the subsequent blocks. Cryptographic mechanisms used by blockchain technology provide integrity and non-repudiation to the system. Additionally, it can help alleviate multiple security risks that threaten traditional centralised systems such as single points of failure. However, since a blockchain-based system has several architectural components, each with complex internal structures, the chance of introducing security vulnerabilities by making poorly informed design decisions is non-trivial. There is a lack of a systematic guide to design secure systems based on blockchain technology [154, 183]. This situation leads to architects operating in an ad hoc manner [127], relying on the wisdom and trust of peers [183]. Moreover, a lack of awareness of security attacks and the consequent impacts on blockchain system architecture can deter practitioners from addressing security issues at the early development stage [155]. As a result, attackers might discover security flaws and breach the system.

The novel contributions of this article are:

- A taxonomy that defines, illustrates, and classifies the key architectural decisions regarding blockchain-based systems including access type, data storage, and transaction computation decisions. This taxonomy is the result of an approach partially guided by a systematic literature review that identifies the major architectural design properties and choices related to blockchain-based systems.
- A mapping approach that associates architectural decisions of blockchain-based systems with potential security attacks and threats. A threat classification model is used to categorise threats associated with the attacks and the architectural choices. Specifically, we use the Microsoft STRIDE threat model [92], because it classifies threats based on the ramifications of their realisation, such as a **denial of service (DoS)**, disclosure of information, or elevation of privilege.

Mapping the proposed taxonomy with the related security ramifications helps software architects to fully comprehend the impact and scope of the security challenges associated with blockchain systems. The security implications of threats can be directly associated with the likelihood and

[1] https://healthlinkages.com/.
[2] https://hashedhealth.com/.

impact of potential attacks on the whole system. This approach provides a basis for evaluating the potential security risks in all dimensions of the system. We advocate security risk assessment at early stages to provide the engineers with more objective guidance that can assist in further analysis for the potential threats and attacks, refining the design and testing for security.

Previous studies have illustrated various security vulnerabilities and the consequent attacks on blockchain-based systems [72, 102, 105]. However, to the best of our knowledge, this is the first study to present a thorough categorisation of threats and their correlation with attacks as well as with blockchain architectural decisions that are susceptible to those attacks. Other studies [169, 180] have presented taxonomies of the architectural properties of blockchain systems and showed the impact of these properties on quality attributes of the system such as performance. However, their effects on security properties—possible attacks and their subsequent threats—have not been covered. Moreover, these prior studies did not systematically determine the major architectural decisions that are considered when designing such a system. To the extent of our knowledge, this work is the first to present a taxonomy that categorises architectural decisions of blockchain-based systems based on a systematic review of the literature.

The rest of the article is organised as follows: Section 2 outlines the procedure followed to conduct our review. Section 3 provides a comprehensive taxonomy of architectural decisions for blockchain-based systems, followed by Section 4, which introduces our mapping approach, and links the classified threats and attacks with our taxonomy. Section 5 demonstrates an application of our work. Section 6 shows the validation of our work and discusses the threats to validity to our approach. Section 7 discusses the related works. Section 8 represents future research directions and concludes the article.

2 RESEARCH METHODOLOGY

2.1 Surveying the Literature

Our empirical route [123] is partially guided by a systematic literature review research method [86]. In this way, we increase the possibility of producing an unbiased exploration of the current body of work on the field and select representative articles that can reflect the major architectural decisions for blockchain-based systems along with their characteristics, attributes, and variants. To achieve this aim, we conducted the following **research question (RQ)**:

RQ1: ***What are the architecturally significant design decisions in blockchain-based systems?*** The search period for this survey starts in 2008, since the original paper describing blockchain appeared that year [120]. We conducted a trial search first to select and refine the query terms used. We tested several possible queries that resulted in either a huge number of papers or research papers that were unrelated to our survey goals. Finally, we settled on *Blockchain AND (Architecture OR Architectural)*. This query has been designed as open-ended to provide exhaustive coverage of the literature where the terms architecture or architectural are often preceded many of the keywords that relate to architecture such as design, tactic, quality, model, or concerns. Searches were performed in five electronic databases: IEEEXplore, ACM Digital Library, Science Direct, ISI Web of Science, and Scopus.

Since not all the resulting papers from the search were related to the research questions, they needed to be filtered first. Hence, we identified several selection criteria that were applied to ensure that the outcomes were objective. The inclusion and exclusion criteria we defined are as follows:

Inclusion criteria (I)
I1: Papers published in peer-reviewed journals, conference proceedings, workshop, or book chapters.

I2: Papers focused on one or more blockchain architectural design decisions.
I3: Papers reporting on fundamental research into software architecture for blockchain and/or their applications.
Exclusion criteria (E)
E1: Papers from disciplines different that computer science, in which they use blockchain merely as a component of the application.
E2: Papers without full text, papers written in other language than English, and duplicate papers.
E3: Papers related to blockchain-based applications without any substantial architectural discussion.

Our study selection procedure employed three rounds to filter the research papers and select the final set.

First round: We selected papers based on metadata, including title, venue name and keywords. In this round, we considered the criteria I1 and E1.

Second round: Authors independently chose papers by reading the abstracts of the papers. In this round, we considered criteria I3, E2, and E3.

Third round: Authors independently chose papers by reading the full text of the papers selected in the previous round. In this round, we considered criteria I2, I3, E2, and E3.

Additionally, we performed a manual search to include relevant papers informed by experience or citations from seminal research. Figure 1 illustrates the search and the selection procedure followed.

Data Extraction. The data extraction process was initially applied to a set of 10 studies that are highly cited and considered the most seminal in the area of blockchain architecture. We read their full text to collect information regarding architecting blockchain-based systems. The considered aspects were: blockchain-based systems *architectural components*, their *variants*, *characteristics*, and *design decisions*. Then, all the extracted information was combined and categorised. We identified the key concepts and dimensions that led to an initial version of our taxonomy. Next, we analysed the full set of selected papers to revise and refine the taxonomy. Therefore, the approach that we followed to build our taxonomy can be described as empirical to conceptual approach. Publication *titles* and *aims* were collected from all included studies.

2.2 Mapping Approach

The second objective of our study is to map the architectural decisions of blockchain-based systems to threats and attacks. To achieve this, we decided to address the following research question:

RQ2: ***How can potential threats and attacks be traced to blockchain architectural decisions and to which components?***
There are various strategies that can be used to investigate ways in which blockchain-based systems may be threatened. We focus on common and widely acknowledged security threats with a malicious purpose that are often posed by cyberattacks, compiled from existing literature. We propose attack and threat classification models with the aim of identifying and tracing the threats to the architectural aspects of blockchain systems. The adversarial tactics categorisation proposed by MITRE [118] is used to classify attacks and the STRIDE threat model is used to classify threats. MITRE provides adversarial tactics and techniques based on community contributions from real-world observations.

We used a general search string *(Blockchain AND ("Security Attacks" OR "Cyberattacks") AND ("Security Threats"))* for identifying blockchain-specific threats and attacks. We have specifically

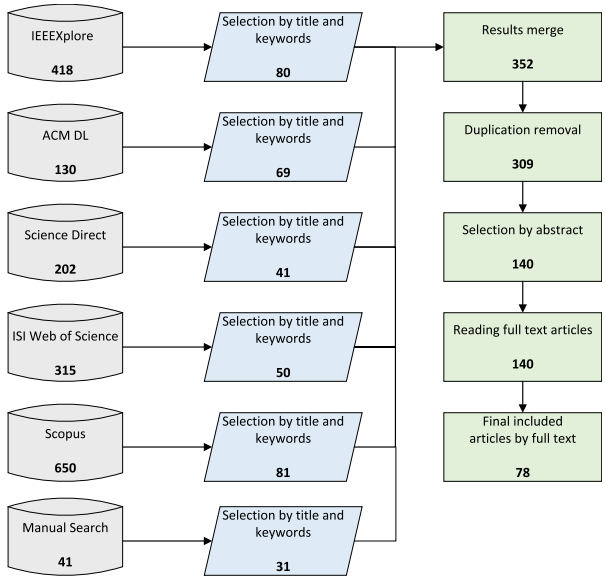


Fig. 1. Search and paper selection procedure.

Table 1. Selected Articles

Publication Type	References
Surveyed and SLR (Secondary study)	[8, 12, 13, 23, 27, 31, 40, 59, 72, 102, 105, 137, 141, 171]
Primary Study	[11, 19, 30, 50, 74, 94, 126, 132, 149, 174]
Grey Literature	[35, 44, 52, 128, 132, 143]

searched for seminal and widely cited surveys and reviews on the topic to help us identify commonly documented blockchain attacks and threats. At the time of writing, we found 14 articles that have surveyed blockchain-specific threats and attacks. Additionally, we have considered 10 primary studies on the topic, where additional attacks were identified. Grey literature is also considered by searching and analysing the first 200 top results from Google. From the selected results, we have filtered the most relevant reports, unpublished research, and articles that relate to the investigation query and published by public or private institutions or organisations. We included those results that identified new/recent issues that were not covered by the standard academic literature. We excluded generic reports related to the blockchain technology without describing specific issues and those that provide superficial and non-elaborating mentioning about attacks and threats. Additionally, when grey literature provides redundant information to that of the peer-reviewed one, peer-reviewed ones were used. Table 1 shows the selected articles. Our search identified 56 distinct blockchain-specific attacks that are defined and classified. Each of the identified attacks and threats can be traced back to its source(s).

Before the classification and mapping process, we first extracted blockchain-specific attacks and threats from the selected articles, along with their definitions, to ensure that all authors had a shared understanding. We have only focused on commonly used strategies for launching the attacks, as attackers may take novel approaches in combining different tactics to launch/execute the attack. The classification was straightforward for the attacks that are not only targeting blockchain, as the MITRE team has classified them under the related tactics. To minimise inherent biases in the

mapping process, two authors with security backgrounds and expertise worked independently on the rest of identified attacks and threats to categorise and map them to blockchain architectural dimensions. Once the authors completed the task, the results were discussed and verified with the third author, and for each disagreement, all authors discussed their rationale to consolidate the results. The final result was reviewed by the fourth author. After analysing the result, we found that several attacks tactics proposed by MITRE are unrelated to any of the attacks in our set. Thus, we excluded these tactics from our mapping. Finally, the result was sent to an expert in blockchain security, who reviewed and provided feedback on our classification.

3 TAXONOMY OF DIMENSIONS FOR ARCHITECTURAL DECISIONS IN BLOCKCHAIN-BASED SYSTEMS

Taxonomies have an essential role in the software engineering discipline, because the categorisation and organisation of the knowledge enables practitioners and researchers to understand and analyse a complex design space and to evaluate and compare design options.

Our work is not only aligned with the academia but also with the industry. The industry has organised concepts, components, models, and other elements around the blockchain technology to facilitate its understanding [170]. The industry has also proposed a playbook of blockchain [7] that defines a process with five phases to assist in the adoption of the technology: (i) problem assessment; (ii) organisational readiness; (iii) technology selection; (iv) blockchain implementation; and (v) blockchain integration. In this context, our work fits in the third phase, since an organisation of architectural decisions in dimensions is intended to support the construction of a platform architecture and an operational model, which are two of the outputs of the referred phase.

As Figure 2[3] illustrates, the results of our survey indicate that the dimensions of key architectural decisions are: (i) blockchain access type; (ii) data storage and transaction computation; (iii) consensus mechanism; (iv) block configuration; (v) key management; (vi) cryptographic primitives; (vii) chain structure; (viii) node architecture; and (ix) smart contract. In the following subsections, we describe each architectural decision from the security perspective and provide a discussion of the quality attribute tradeoffs entailed by each architectural choice. Table 2 illustrates these attributes.

3.1 Blockchain Access Type

Among the crucial design decisions for blockchain-based systems is the choice of blockchain access, which identifies who is permitted to participate in the network and in what transactions. Blockchain access is classified into three categories: public, private, and consortium [179]. Each has its own security properties and limitations.

3.1.1 Public Access. In this access type, any node can read, send, and verify transactions. This type is often known as a permissionless digital ledger, where nodes are also able to create new blocks of transactions [112]. A public blockchain is fully decentralised; there is no central authority that controls the system [98]. Additionally, information is distributed, shared, and recorded in all nodes that participate in the network. Therefore, this type is widely available, since they have no single point of failure. Importantly, immutability and integrity of information are also supported in public blockchains, as any node can verify that the data has not been tampered with and, once the information is written in the block, it cannot be altered without detection, because it is stored in different nodes in the decentralised network [90]. The level of transparency of this type of blockchain is high as well, since the records and their updates are available to the public.

[3]The diagram follows an extended feature model notation.

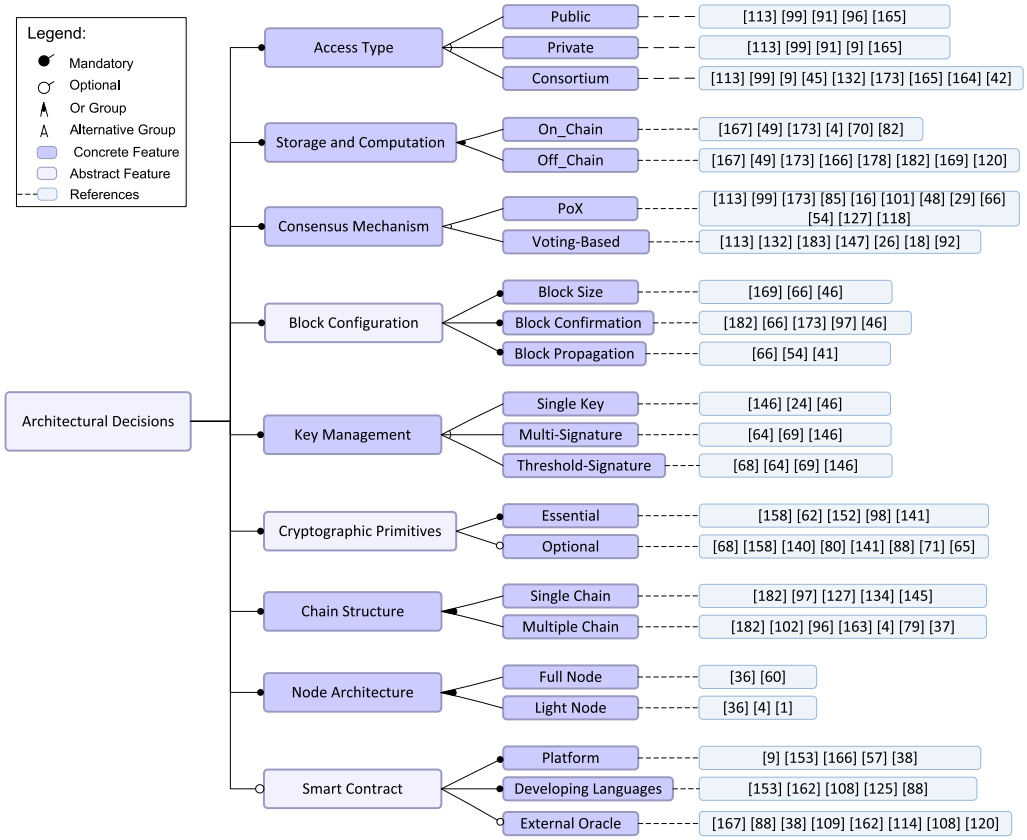


Fig. 2. Taxonomy of major dimensions for blockchain architectural decisions.

While public blockchains are highly transparent and the chained information is visible to other peers, privacy is difficult to achieve; extra cryptographic mechanisms are required to strike a balance between transparency and privacy. Hence, the decision for adopting this type of blockchain requires architects to consider the tradeoffs between privacy and transparency in the given context.

3.1.2 Private Access. In this type, often known as permissioned blockchains [9, 112], only one organisation has written permissions, while read permissions can be public or restricted to a preselected set of readers [98]. Since private blockchains are controlled by a single group, they are known as centralised blockchains. In this type, all validators are known, as they are all members of a single group—and a centralised authority controls the network by verifying each node, and allowing or rejecting requests to join the network. These specific features of private blockchains give them some security advantages over the public type. Private blockchains provide a greater level of privacy [90], especially when read permission is also restricted [178]. However, private blockchains require trust in identities, especially when the number of nodes in the network is low, as the parties might act in collusion to threaten the system. Incorrect trust assumptions, when selecting participants in the network, can have security ramifications. Moreover, as public verifiability is not required, the integrity of the system can only be ensured if the system is not breached.

Table 2. Quality Attributes per Dimension of Architectural Decisions

Dimensions of Architectural Decisions		Quality Attributes	
		Upside	Downside
Access Type	Public	• Fully decentralised • Widely available • High immutability • High integrity • High level of transparency	• Low privacy • Low confidentiality
	Private	• Centralised administration • High privacy	• Low integrity • Low immutability
	Consortium	• Partially decentralised • Better availability than in private ones • Better privacy than in public ones	• Low integrity • Low immutability
Storage and Computation	On-Chain	• Based on the applied blockchain access type	
	Off-Chain	• High privacy • High confidentiality • Reduced cost • Low latency	• Low availability in case of a centralised administration • Low integrity • Low immutability
Consensus Mechanism	Proof-based (PoX)	• Unlimited number of nodes • High decentralisation • Free join • Award • Low energy consumption in case of PoS	• Mismanaged node identity • High energy consumption in case of PoW
	Voting-based	• Managed node identity • Low energy consumption	• Not free join • Mostly no award • Limited number of nodes • Low decentralisation
Block Configuration	Block Size	• Higher throughput in case of large size	• Network congestion and slower propagation in case of a large size
	Block Confirmation	• Low latency in case of fast confirmation	• Increase fraud attempts in case of fast confirmation
	Block Propagation	• Based on the block size	
Key Management	Single Key	• Better accessibility than in other types • Low latency	• Low availability • Low integrity • Repudiation
	Multi-Signature	• High integrity • Non-repudiation	• High latency • Low confidentiality
	Threshold-Signature	• High confidentiality • High integrity • Non-repudiation	• High latency
Cryptographic Primitives	Essential Primitive	• Hash: immutability, integrity, high efficiency • Digital signature: authentication, non-repudiation, medium efficiency	
	Optional Primitive	• Confidentiality, privacy, anonymity, unlinkability	
Chain Structure	Single Chain	• Manageability • Low privacy	• High latency • Low scalability • Low throughput
	Multiple Chains	• High privacy • Better scalability and throughput than in single chain	• Complex to manage
Node Architecture	Light Node	• Low storage cost	• Partial dependency on full nodes
	Full Node	• High availability	• Costly
Smart Contract	Platform	• Based on the applied blockchain access type and consensus mechanisms	
	Developing Languages	• Based on the applied blockchain access type and consensus mechanisms	
	External Oracle	• High availability and trustworthy in case of a decentralised oracle • High efficiency in case of a centralised oracle	• Low availability and low trust in case of a centralised oracle • Low efficiency in case of a decentralised oracle

3.1.3 Consortium Access. This type of access is partially decentralised, meaning that rather than the system being controlled by a single organisation, a group of pre-selected nodes from multiple organisations is responsible for consensus and block validation [45]. Read and write permissions can be determined by the consortium; they can be public or limited to selected nodes in the network [112, 131]. This type of access provides a balance between public and private. Since it is partially decentralised, it provides better availability than a private access, while it has better privacy than the public one, as it is partly private. The possibility of tampering, however, is one of the significant security limitations of this type of access. Since the chain is controlled by a group of nodes, they can collude and alter or reverse the transactions, which negatively affects the immutability promise of the technology. *Hybrid* is a type of blockchain that combines the features of public permissionless and private permissioned classes [163]. Participants can manage the access

feature and decide who gets access to which data on the blockchain. This allows systems to operate with transparency without having to reduce system privacy [42].

Recognising the security strengths and possible risks of each type of blockchain access assists architects to choose the model that best matches the requirements of the systems they are attempting to build. In practice, an access type that is optimal for a particular case, such as financial systems, may be sub-optimal for another case, such as healthcare systems. In Reference [164] authors provided a methodology to determine the appropriate access type for a particular blockchain systems scenario. Additionally, References [112] and [90] explain the differences between each type.

3.2 Storage and Computation

Blockchain technology choices have implications on storage space and computation. Choosing to place data and computation on-chain or off-chain is a critical architectural decision that involves several tradeoffs such as cost, privacy, and integrity of information.

3.2.1 Data Storage. Several properties must be considered when deciding to store data on- or off-chain, such as scalability, performance, privacy, and confidentiality [166]. Confidentiality and privacy of sensitive information stored on a public blockchain cannot be guaranteed, as the content is visible to every node joining the chain. In some applications, data are required to be visible to specific nodes, not to all the nodes in the blockchain network. In this case, storing data off-chain can be helpful to overcome and mitigate such limitations [172]. Commonly, when there is a decision to include off-chain storage, it will be used to store raw data, while meta-data and hashes of raw data will be stored on-chain. A set of on-chain data management patterns have been proposed in Reference [168].

The decision to select off-chain storage needs to be taken deliberately, because inadequate analysis of the security consequences of any storage type can lead to security risks. For example, an architect might decide to use a centralised solution, such as a private cloud storage, as it is easy to configure and manage; yet this solution could become a single point of failure. Moreover, if the raw data that is stored off-chain is deleted or lost, it cannot be recovered from its hash value, which is permanently stored on–chain. Another option would be to use a peer to-peer decentralised file sharing platform such as **IPFS (InterPlanetary File System)** [15, 165, 177], Swarm [71], OrbitDB [125], or Filecoin [129]. Due to the decentralised nature of these kinds of storage, if one node disconnects, the data can still be accessed. One of the drawbacks of off-chain storage is that the integrity of the raw data is based on the soundness and the security of the hash algorithm that is applied to hash the raw data, as Section 3.6 discusses.

3.2.2 Transaction Computation. Transaction execution, validation, and consensus mechanisms in blockchain increase the response time, as they require communication and execution overheads. Moreover, mining processing—the process of assigning new blocks to the chain—is expensive, because it typically involves a fee. Some blockchain applications require micropayment transactions, payments of small amounts of money, often just a few cents. This kind of transaction is very costly to be done on-chain, because the transaction fee that is required to execute the transaction might be higher than the cost associated with the transaction. As it is infeasible to execute and store each micropayment transaction on-chain, several applications that require such kinds of transaction have been established to create a separate off-payment channel between the participants [181]. This construction helps to reduce latency and cost and increases throughput. The constructions are also commonly known as Layer-2 channels or state channels.

There are several protocols available for implementing off-chain payment channels, each of which has its own advantages and disadvantages. Lightning Network is a protocol that allows the routing of payments through several intermediary nodes. This approach reveals information

about the nodes and the performed transactions, as the intermediary nodes can see the flow of funds through the channel. To mitigate this problem, Bolt protocol has been proposed [69]. This protocol includes a set of techniques for building anonymous payment channels. There are also other methods that have been proposed to preserve the privacy of users of the off-chain channel. A comprehensive analysis of the off-chain channel and its related protocols can be found in Reference [81]. A set of off-chaining patterns were also proposed in References [49, 119]. The objective of these patterns is to move computation and data off-chain, without compromising blockchain features such as trustlessness.

3.3 Consensus Mechanism

While blockchain is a decentralised technology that relies on a decentralised authority to manage, authorise, and verify the transactions, a fault-tolerant consensus protocol is required, which is a set of rules to assure that all nodes agree on the new block that is appended to the blockchain. Transactions verification and immutability depend on the selected consensus mechanism. There are several consensus mechanisms in use in the existing blockchain technologies. One crucial point is to understand the type of faults and tradeoffs relating to the application domain to select an optimal consensus protocol that helps to secure the blockchain. When a sub-optimal protocol decision has been taken, replacing the consensus protocol in blockchains will be challenging, requiring a serious code rewrite. Researchers have suggested that a general-purpose permissioned blockchain should be built with pluggable consensus mechanisms, and it has been emphasised that there is no "one-size-fits-all" solution [152]. Hyperledger fabric is the first blockchain that has come with a pluggable (not hard-coded) consensus protocol.

A consensus protocol inside a blockchain can be classified into two classes [122]. The first class is the proof-based consensus mechanism, also known as **Proof-of-X (PoX)**. This type of mechanism is preferable when the expected number of nodes joining the network is large as in the case of public blockchain. The second class is a voting-based consensus mechanism, also known as **Byzantine Fault Tolerant (BFT)**-based consensus. This type is more applicable to consortium and private blockchains where the number of nodes is often restricted and all the nodes inside the network are known and adjustable. However, PoX is also applicable to the consortium and private blockchains and is not restricted to public ones.

The core concept of PoX consensus is to give a right to the node or set of nodes that show or accomplish a specified proof to be allowed to append a new block to the chain and receive a subsequent incentive. Several variants of this consensus mechanism have been proposed in the literature. The main one is **Proof-of-Work (PoW)** [65], which was proposed when Bitcoin appeared in 2009. This consensus type requires solving a puzzle with adjusted difficulty, demanding high computational power consumption. The solution of the puzzle involves a group of nodes; the nodes that first solve the puzzle are then allowed to broadcast their blocks to the blockchain network. The second popular protocol in this type of category is **Proof-of-Stake (PoS)** [16, 84], which was proposed in 2011 and claimed to mitigate the high resource consumptions and the limitations of PoW. The main idea of this consensus type is utilising stake to determine the possibility of nodes to mine the subsequent block of the chain. Hence, the nodes with a higher stake have a higher chance of validating and broadcasting the block. Although PoS requires lower energy consumption than PoW, the latter provides better decentralisation [21]. Often, the number of miners in the PoW is much larger than the number of validators in PoS. There are several variants have emerged to tackle the drawbacks of PoS, including **Delegated Proof-of-Stake (DPoS)** [100], CloudPoS [21], and **Proof-of-Supply-Chain-Share (PoSCS)** [21]. In addition to PoW and PoS, there are multiple consensus protocols that can be classified under PoX category, including Proof-of-Space [48], Proof-of-Activity [17], Proof-of-elapsed time [29], and more [117, 122, 172].

The main concept behind a voting-based consensus mechanism is that agreement to append the block to the chain is based on the majority of node decisions. In particular, K nodes, where K is a given threshold, are required to show the same proposed block before accepting it in the chain. Most of these algorithms require at least one process to receive and validate the votes from all other processes and then broadcast the result. There are several consensus protocols proposed under this category, such as **Practical Byzantine Fault Tolerance (PBFT)** [26], which has been used by the Hyperledger blockchain platform, Ripple [142], R3 Corda [34] with BFT-SMART [18], Quorum with Raft [91], and more [122, 131, 146, 182]. More about consensus algorithms can be found in References [122, 158].

3.4 Block Configuration

There are three main aspects to be considered regarding block configuration: block size, confirmation, and propagation.

3.4.1 Block Size. This refers to the maximum number of transactions aggregated within a block. The optimal block size is still a debatable subject. The system's throughput is sensitive to the size of the block, as increasing its size is a way to enhance the throughput of the blockchain system. However, arbitrary increases in the block size without carefully analysing the consequences of this increase can adversely affect the system. Large block sizes can cause network congestion and slower propagation speeds that, in turn, result in raising the number of stale blocks [65]; these are blocks that are not joined to the longest chain because of a conflict or concurrency, which leads to security risks. A tradeoff between security risks and throughput requires an analysis at an early stage when deciding on the block size. Alternative solutions need to be taken into account to enhance the throughput of the blockchain systems, such as second layer payment channels [168].

3.4.2 Block Confirmation. Commonly, in the blockchain-based systems, a transaction is confirmed after waiting for a certain period, which can be a specific number of blocks that have been created once the transaction has joined the blockchain. Deciding on the required number of blocks for confirmation is a critical design decision. This strategy has been used to guarantee that a transaction is attached to the longest chain securely. However, researchers [172, 181] have argued that real-world businesses often require an immediate response, as no one wants to be at risk of losing assets during the waiting time. Therefore, an immediate confirmation has been proposed in some consensus algorithms such as PBFT and Proof-of-Familiarity [172]. Another strategy for transaction confirmation is to add a checkpoint to the blockchain [169]. The transaction is accepted once the checkpoint is valid; otherwise, if the fork chain starts before the checkpoint appears, then it will be rejected by all nodes.

3.4.3 Block Propagation. In a blockchain network, a broadcast protocol is needed to distribute blocks to the peers in the network. The architectural decision of the underlying broadcast protocol affects the security, reliability, and scalability of the network [54]. Several protocols have been proposed to deliver blocks to the nodes in the network. Advertisement dissemination [65] is one of the common protocols that has been used by the PoW blockchain. In this protocol, if node A receives a new block from another node in the network, then it advertises the hash of the block to its other connections. If one of the nodes, e.g., node B, has not previously received this block, then B will demand it from block A, which will then send the contents of the block to B. Other propagation mechanisms are proposed, such as send header, unsolicited block push, and relay networks, with the aim of reducing the propagation delay, as blockchain forks are caused due to long propagation time [65]. Obviously, there is a correlation between the propagation latency and the size of the block, as a large block is propagated slowly in the network, which allows an adversary to leverage

this delay. Multiple ways have been introduced for enhancing the propagation of the blocks such as minimise verification, pipelining block propagation, and connectivity increase [41].

Block configuration aspects are often based on the blockchain platform. At the time of writing, the average Ethereum block size is 20 to 30 KB, transaction confirmed every 15 seconds, and advertisement hybrid propagation mechanism is used for block propagation. Therefore, it is essential that the developer knows these details to select an appropriate platform for their system.

3.5 Key Management

This section explains the various ways for signing transactions and different options for storing the users' keys.

There are alternative signature schemes to sign the transaction in a blockchain: using a single key, a multi-signature scheme, or a threshold signature scheme. Storing and depending on only a single key to sign sensitive transactions, such as financial or cryptocurrency transactions, introduces a single point of failure, which contradicts decentralisation and the distributed trust concepts of blockchain technology [46]. To mitigate this threat a multi-signatures mechanism was proposed [67]. This mechanism requires multiple secret keys to generate the signatures, and M signatures for N private keys are required to sign any transaction. A simple example is a two-factor wallet [114] that requires two devices, such as the user's mobile phone and laptop, to sign any transaction. However, this scheme increases the transaction size linearly with the number of signatures, which subsequently increases the transmission time. Additionally, this scheme negatively affects the confidentiality of the transaction, since it will be visible in the public block that a multi-signature transaction has been used.

Threshold signatures are an alternative signature scheme where the transaction can be signed using shares of a single private key. These shares are split among N parties using threshold cryptography. This scheme provides the same M-of-N security but increases the confidentiality, since transactions are indistinguishable from non-threshold transactions on the blockchain and the parameters M and N are kept private. ECDSA threshold-signature has been introduced by References [63] and [68] to enhance Bitcoin security. Other research [145] proposed RSA and BLS threshold signatures for the Hyperledger Fabric blockchain platform. The security of the threshold signature scheme is based on a cryptographic algorithm that is used to apply the digital signature, as described in Section 3.6.

There are several alternative ways to manage and store private keys, each of which has its own security implications [137]. In most blockchain applications users use a piece of software, called a wallet, to store their private keys securely. Public keys and associated addresses can also be stored in the wallet [24]. Keys can be stored in an off-line wallet, which is the most secure type; however, it is inconvenient to use, and it is commonly used as a backup. Alternatively, online wallets can be used, which is more convenient, but the server can steal such keys. A local or device wallet is another kind of wallet where the keys are stored directly in the specific file; thus, users can have full control over their keys.

3.6 Cryptographic Primitives

Cryptography is a key component of blockchain technology, as the security of the whole blockchain system is based on the security of its underlying cryptographic primitives. Compromising one of them adversely affects the security of the entire blockchain system. Cryptographic primitives in blockchains are classified into: essential and optional.

3.6.1 Essential Cryptographic Primitive. This type includes hash functions and digital signatures.

Cryptographic Hash Function: In blockchain, the hash function is used in many operations such as addresses and block generations, message digests in signatures, and in some consensus mechanisms such as PoW. A secure hash function should be collision-resilient, tamper-resilient, and should be a one-way function, where its result should be easy to verify and hard to invert [157]. These proprieties provide two security attributes to the blockchain system: immutability and integrity. **Secure Hash Algorithms (SHA256)** and RIPEMD160 [140] are popular hash functions in blockchain and have been used in most blockchain platforms [157]. However, there are also several hash functions that have been used in different platforms, such as Ethsh [151] in Ethereum, SCrypt in Litecoin [56], and other platforms. Based on Reference [89] results, SHA256 and RIPEMD160 are among the fastest algorithms and have the best performance. This is because these algorithms can validate blocks without occupying a lot of memory space and processing power. As Reference [157] stated, the efficiency of hash algorithms is the highest comparing to other type of cryptographic primitives. In References [89, 157], authors comprehensively analysed the performance of hash functions suitable for use in blockchain.

Digital Signature: Asymmetric-key cryptography primitive is used to generate a digital signature, where each node should have a pair of a private key and a public key. The private key should be kept secret, since it is used to sign the transaction, while the corresponding public key can be used by any node in the system to confirm the ownership of the signed transaction and to verify that the transaction has not been modified or tampered with. One of the security properties that secure digital signatures provide is authentication where a valid signature indicates that the transaction is signed by the known user. Non-repudiation is another security property where the node that sent the transaction cannot deny it. Moreover, a valid signature can guarantee the integrity of the transaction and that it has not been altered in transmission. The key generation algorithm of a digital signature scheme should have a good randomness source to generate different key pairs for different users. A weak randomness source could allow an attacker to recover the user's private key and sign transactions. In Reference [115] a vulnerability is reported in an **elliptic curve digital signature algorithm (ECDSA)** [140], which is used in the most common algorithm in several blockchain platforms, such as Bitcoin and Ethereum. This algorithm fails to generate enough randomness during the signature process, which allows an attacker to discover the user's private key. However, based on Reference [55] analysis, ECDSA requires lower computation and it is much faster than other type of digital signature, including RSA and DSA. These two algorithms require longer keys to provide a safe level of encryption protection compared with ECDSA, which requires much shorter keys and that leads to much better performance. As stated in Reference [157] the efficiency of digital signature is medium comparing to other type of cryptographic primitives. Alternative digital signatures, listed and explained in detail in Reference [157], are also used by several blockchain platforms.

The choice of secure cryptographic primitives is essential, as a vulnerable one can weaken the entire blockchain system. The main cryptographic schemes that are used in blockchain, including ECDSA, depend on the difficulty of prime factorisation and the discrete logarithm problem. However, these schemes can be threatened by applying quantum algorithms that work in a polynomial time to break such schemes. Thus, using quantum-resistant cryptography and post-quantum mechanisms, which utilise cryptosystems that stay secure under the assumption that an attacker is in possession of a large-scale quantum computer, are recommended by References [62, 97, 174]. However, the authors in Reference [85] argued that the robustness of these algorithms are based on unproven assumptions, and they are computationally intensive. They suggested the use of **quantum key distribution (QKD)**, which provides security based on the laws of quantum physics where a secret key is distributed using the quantum channel. As quantum physics is fundamentally

random [62], the bit stream generated by a **quantum random number generation (QRNG)** is provably random. Thus, this ensures the generation of truly random encryption keys. A thorough analysis of QKD and QRNG can be found in Reference [85].

3.6.2 Optional Cryptographic Primitive. This includes the symmetric algorithms and other cryptographic primitives that are used mainly for enhancing the confidentiality, privacy, anonymity, and unlinkability of blockchain-based systems. One or more of these properties might require to be provided by the system under consideration, especially in the case of a public blockchain, as the chain's state is transparent, and everyone can access the chain without restriction. Therefore, cryptographic primitives that preserve identity and transaction privacy need to be considered in the decision-making process.

There are several cryptographic methods that aim to protect identity and transaction privacy in the blockchain, including **zero-knowledge proof (NIZK)** [70] and Homomorphic encryption [64]. Zerocash [139] is a privacy protocol that makes use of NIZK proof and homomorphic scheme to achieve both anonymity and transaction privacy. However, it involves high computational costs for generating the transaction proofs. Lelantus [79] protocol has emerged to enhance the confidentiality and privacy of the blockchain and mitigate the disadvantages of zerocash. Multiple privacy protocols leveraging one or more of the cryptographic schemes have emerged [87]. Authors in Reference [157] systematically discussed, analysed, and compared the cryptographic schemes.

3.7 Chain Structures

Creating single chains or multiple chains is one of the essential architectural decisions to design a secure, reliable, and efficient blockchain-based system.

3.7.1 Single Chain. All transaction types generated in a blockchain-based system are recorded together in a unique chain when this type is selected. Clearly, a single chain is easier to manage; yet it increases latency and negatively affects the scalability and the throughput of systems [167], especially when a sub-optimal data structure is implemented. In a classical blockchain such as Bitcoin, the data structure is a linked list of blocks that creates a chain and, when a conflict appears, the longest chain is selected by the nodes in the network. This selection rule increases the chances of double spending threats. The **Greedy Heaviest-Observed Sub-Tree (GHOST)** protocol was proposed by Reference [144] and has been used by Ethereum, but it changed the selection rule to select the side whose subtree has the most work. This protocol enhances security by increasing mining fairness. **Directed Acyclic Graph (DAG)** is an alternative structure that has been proposed by Reference [96] to enhance the security of a traditional blockchain structure. They changed the structure from a list to a graph where each block references multiple predecessors. This structure is better suited when the block size is large or when blocks are frequently created. Analysis showed that this structure considerably enhances the mining power utilisation [96]. TrustChain [126] is another data structure that can be used in permissionless blockchain systems. This structure includes a NetFlow algorithm that calculates the trustworthiness of the nodes using the prior transaction as input. This algorithm makes blockchain systems more secure, as it prevents untrusted fake identities from joining the network. There are several data structures that have been demonstrated in the literature, including LeapChain [133] and segregated witness [169].

3.7.2 Multiple Chains. Instead of storing and executing all types of transactions and information on one chain, information and the transactions can be classified, executed, and stored in more than one chain. There are several structures involving multiple chains that have emerged to overcome scalability and throughput issues in blockchain. The sharding scheme, also known as a layer-1 scalability method, is one of the solutions aimed at improving the performance of the blockchain.

It does this by splitting the processing of transactions among smaller sets of nodes, called shards. These shards operate separately and in parallel to enhance throughput and decrease storage and computation overheads. However, the possible corruption of the shards is one of the security issues that needs to be tackled, in that malicious nodes can easily dominate a single shard. To tackle this concern a sharding scheme requires the random division of the network into small shards to prevent any shard from accepting an overwhelming number of adversaries. Ethereum 2.0 and RapidChain [175] are sharding mechanisms that randomly attach nodes to specific shards and regularly reassign them at random intervals. A detailed explanation and systematic analysis of several sharding protocols can be found in Reference [37].

Another scalability solution is called sidechain [109], which is an independent blockchain that has its own ledger. However, it is not a standalone platform, as it is linked to the main chain to allow the assets to flow from one chain to other. This scheme can extend the blockchain to support multiple applications without increasing the load on the main chain. Satellite chains [101] are types of multiple chains, where each interconnected but independent sub-chain can have a different consensus mechanism running privately in parallel in single blockchain system. Double chain [95] is another instance that aims to ensures the privacy of the user information by storing users' information in one chain, while transactions are stored in another chain. Multiple chain structures are intended to enhance performance and privacy of blockchain systems.

However, the nodes classification in this type of chain might be complex, and it requires a specific management strategy [78]. Additionally, not considering secure protocols for exchanging information, assets, or tokens among different chains can weaken the security and soundness guarantees of blockchain systems. **Unitary Interchain Network Protocol on Transport Layer (UINP)** supports cross-chain [104, 162] schemes from the transport layer. This protocol gives low latency convenience and provides high security similar to **Transport Layer Security (TLS)** [162]. UINP employs point-to-point data transmission that can match transactions without requiring a third party for validation.

3.8 Node Architecture

Blockchain is a peer-to-peer decentralised network where the participant nodes can be full nodes or light nodes. The job of full nodes is to keep and verify a local copy of the entire chain of transactions. Each full node can access all the historic transactions and verify if the new one is valid and consistent with the existing transactions. As all records are replicated in each full node, blocks and transactions can be verified locally if one of the full nodes is down the blockchain network is not affected because there are other active full nodes in the network [20]. These properties contribute to the availability of the blockchain network and reduce a single point of failure threat. Pruned node is a full node that erases some data when it reaches a particular limit to allow the new blocks to be stored and preserve blockchain size [60]. Even though having full nodes adds robustness to the blockchain network by omitting the need for a centralised party, this type of node requires a substantial amount of storage and processing cost to download a full copy of the entire chain and to execute the verification operations. Since often the size of the blockchain increases linearly due to the immutability and append-only properties, the overhead of the full nodes continually grows. As a result, low-end users, who have mobile and smart devices, may be reluctant to participate in a blockchain.

To overcome the aforementioned drawbacks and allow end-users with limited resource devices to join blockchain systems, light nodes were introduced [120]. In this type of node, downloading and verifying the full set of blockchain transactions is unnecessary, as the light nodes only need to store block headers that are requested from a full node to employ transaction verification. **Simple Payment Verification Protocol (SPV)** can be utilised by light clients. They request the block

header from a full node that, in turn, provides them with the required information. However, requesting a block header from a single full node introduces a serious security risk, as the full node can behave maliciously and provide a fake header to the light nodes. To mitigate this risk, light nodes can request block header from multiple full nodes and then compare the received results. This approach adds overheads to the light node, as it needs to establish a secure connection to each full node and that can slow the verification operation and add complexity. There are several strategies presented to address this issue, such as the **Distributed Lightweight Client Protocol (DLCP)** [36]. In this protocol, the request can be encrypted before sending it to a predetermined number of full nodes that can access and process it and then relay it to the light client with a single response. Then the light client is required to decrypt the response only once for verification, irrespective of the number of contributing full nodes to ensure that all full nodes agree in one response. This approach reduces computing and communication complexity on the client's side and provides lower latency.

There are several protocols that have been proposed to mitigate the computation overhead and storage size from the client's side, such as Blockstack [4], **distributed hash tables (DHTs)** [1] and more [36]. The architects need to investigate and analyse the security properties of these protocols to mitigate the potential risks, especially if the application domain needs a blockchain with hundreds of gigabytes in size and lightweight users are participating in the blockchain application.

3.9 Smart Contracts

Smart contracts offer a general-purpose computing platform to provide more complex programmable transactions [72, 102]. A smart contract is a decentralised piece of code designed to impose the negotiation of a contract's instructions automatically without the need of a central authority to approve it. Importantly, once the contracts are deployed, they cannot by modified and the author of the contracts will not have any control over them. Thus, a cautious approach is required when designing and structuring smart contracts.

The reliability and security of smart contracts are based on the consensus mechanism and the decisions of the underlying blockchain platform, programming language, and including an external oracle. Selection of blockchain platform depends on the access type that the developer decides to choose. There are multiple platforms for developing smart contracts, each of which provides various features. In Reference [57], the authors provide a decision module to assist the developers in selecting the suitable platform based on its criteria and quality attributes. Each platform supports one or more than one programming languages. Some platforms, such as Hyperledger fabric, support general purpose programming languages such as Java, Python, and GO [9, 135]. As claimed in Reference [152], using such a language facilitates and accelerates the development process, as the developers are not required to learn a domain-specific language and they can continue with familiar languages. Some applications require domain-specific languages [106, 124], such as Solidity [39] and Vyper [153], the two most active and maintained languages used in the Ethereum platform, to enhance the security of smart contracts, making them like traditional contracts and more straightforward to understand. In References [38] and [135], the authors provide a comparison of the available platforms and discuss the supporting languages. Considering security patterns [107, 161] and adhering to best practices [43, 66] are also crucial to guarantee the correctness of the contracts.

One of the critical design decisions related to smart contracts is introducing an external oracle to the isolated blockchain environment [169]. While smart contracts cannot access any data from outside the blockchain environment, a trusted third-party oracle can provide the contracts with the required information. Blockchain oracle is an external data agent that accesses real-world data and transmits it to the blockchain to be leveraged by smart contracts [161]. Moreover, the

external oracle role is not restricted to fetching the data from outside of the blockchain, but it can ensure the validity and the authenticity of the fetched data to guarantee a valid execution of the contracts [113]. Blockchain oracles, such as Provable [130], can retrieve data from a centralised server. The efficiency of this type is high, but a single point of failure is introduced, which might affect availability and accessibility to the data. A distributed type of oracle, such as ChainLink [51], resolves these issues, as it contains several redundant oracle servers. These servers are trusted by the whole blockchain network and do the same job of checking the external state. However, the efficiency of this type is low, as it leads to higher latency for data processing [108].

4 MAPPING THREATS AND ATTACKS WITH BLOCKCHAIN ARCHITECTURAL DECISIONS

Even though the security properties of blockchain technology help to make the system based on it resistant to some kind of attacks, they do not make it completely immune. This is because the system may be subject to a number of security threats if inappropriate architectural design decisions are made.

The architecture of a software system is the set of structural design decisions that serve as a blueprint for the construction and evolution of the system [14]. Decisions that transversely impact the system include the selection of technological platforms, the selection of structural components, and quality attributes (e.g., security, performance) [134]. In particular, architecture-related security concerns can be fixed more efficiently if they are identified and assessed at an early stage.

Multiple studies have shown how weaknesses in a software system's architecture may have a greater influence on numerous security concerns that allow adversaries to attack the system [76, 77]. Schemes for architectural security analysis have also been proposed to identify potential attacks and threats before the system is developed [6]. Additionally, our previous work [2] proposes an approach to assess smart contract security design weaknesses. This increases the awareness of the developer of the potential security issues before publicly deploying the contract.

There are multiple examples of successful attacks that have been facilitated because of making sub-optimal architectural design decisions. In July 2020, an unauthorised third party accessed the e-commerce and marketing database of the Ledger website company. This cyberattack led to a massive data breach that allowed the scammers to apply phishing attacks trying to trick users into revealing the keys to the crypto wallet [93]. Deciding to store sensitive data into an insecure off-chain centralised database facilitated the success of these attacks.

Making use of threat modeling is a crucial part of the development process when it comes to enhancing the security of the system. Microsoft reported that security vulnerabilities significantly decrease after including threat analysis in the development process [148]. *Threat modeling* [159] is a systematic analysis of the design of a system. This analysis helps to identify, rate, and prioritise design-level security threats. This assists in focusing on first addressing threats that represent the highest risk by following appropriate mitigation strategies. Threat modeling involves a structured method that is more cost-efficient and effective than conducting security analyses in a haphazard manner without recognising distinct threats in each architectural component of the system.

In this study, we use the STRIDE threat modeling classification approach proposed by Microsoft to classify different threat types into six categories, as shown in Figure 3. This approach classifies the threats based on the implications of their realisation, such as the manipulation of information, denial of service, and elevation of privilege. The ramification of threats can be mapped to the impact of their incidence, as such mapping is crucial for the assessment of the security risks in the blockchain systems. Each class of the STRIDE model covers the unique sort of attacks that lead to a specific type of threat. Noticeably, one attack can pose several threats in such a threat classification; for instance, a majority attack—also known as a 51% attack—can pose multiple threats, such as

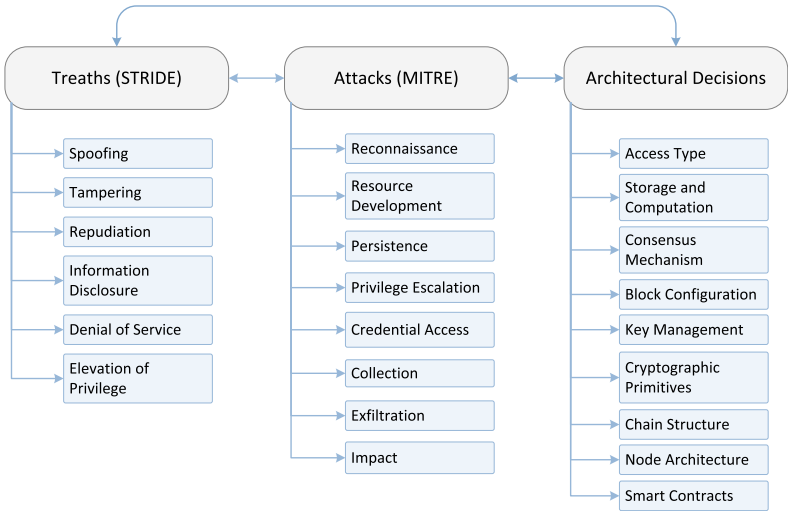


Fig. 3. Mapping threats, attacks, and architectural decisions.

tampering with transactions, disclosing sensitive information, and/or the elevation of privilege [121]. The STRIDE threat model has been used in the blockchain context by Reference [83] to classify and analyse the risks associated with blockchain-based records management. However, to the best of our knowledge, we are the first to identify and classify the threats at a low-level of the blockchain-based systems to link them to specific architectural decisions.

We categorise the possible attacks with regard to blockchain systems based on the adversarial tactics categorisation that proposed by MITRE [118]. Each tactic characterises a high-level description of an attack behaviour. Previous studies have categorised the attacks in terms of organisation and accessibility point of view [72, 102], and some have illustrated possible attacks for specific blockchain applications [33, 59]. To the best of our knowledge, ours is the first study that attempts to shed light on not only the attacks, but also on the threats posed by each form of attack.

4.1 Attacks and Threats Classification in Blockchain-based Systems

This section shows how an architect can use the categorisation of attack information to identify threats in a blockchain-based system by considering the following steps: (i) determine architectural dimensions of the blockchain based-system and how each attack category can breach them (Section 4.1.1); (ii) determine the threats that affect the architectural dimensions of blockchain-based system (Section 4.1.2); and (iii) determine the security threats caused by each attack category using the STRIDE threat model (Section 4.1.3). This way, an architect can map all applicable attacks and their associated threats in the blockchain system. Figure 3 captures the essence of the mapping.

4.1.1 Linking Attack Categories with Blockchain Architectural Decisions. Attacks that target blockchain are aggregated and classified based on MITRE's attack tactics. In this study, we select attack tactics categories that are applicable to blockchain systems. Each tactic represents the objective that adversaries attempt to accomplish. Table 3 shows attack tactic categories and the related techniques that each architectural dimension of a blockchain-based system might be prone to. The selected attack categories are illustrated with examples of attacks that are applicable to the blockchain systems as follows:

Table 3. Linking Attack Categories with Blockchain Architectural Decisions

Attack Tactics	Blockchain Architectural Dimensions								
	Access Type	**Storage and Computation**	**Consensus Mechanisms**	**Block Configuration**	**Key Management**	**Cryptographic Primitives**	**Chain Structure**	**Node Architecture**	**Smart Contracts**
Reconnaissance	*Public:* Deanonymisation attack [19], Replay attack [137]; *Private:* MITM [3, 50]; *Consortium:* MITM [3, 50]				*Single-key:* Malware attacks [40], MITM [3, 50]; *Multi-Signature:* Malware attacks [40], Deanonymisation attack [19]; *Threshold-Signature:* Malware attacks [40]				Malware attacks [40], Replay attack [137]; *Solidity:* *Solidity:* Overflow/ underflow attack [137]; *Hyper-ledge Platform:* Range query risks [23], Log injection attack [23]; *External oracle:* MITM [3, 50]
Resource Development	*Public:* Sybil attack [126], Majority attack [102], Timejacking attack [137]		*PoW:* Majority attack [102], Selfish mining [102], Vector76 attack [72]; *BFT:* Consensus 34% Attack [35]; *PoS:* Majority attack [102], Long-range attacks [72], Short-range attacks [72]; *Vote-based:* Whitewashing [40], Hiding block attack [40]	Block withholding [137], Finney attack [3]			Block withholding [137], Selfish mining [102], Vector76 attack [72], Finney Attack [3]		*EOS Platform:* Roll back attack [143], Replay attack [143], RAMsomware attack [94]; *External oracle:* Oracle Manipulation Attack [132]
Persistence	*Public:* Majority attack [102], Sybil attack [126]		*PoW:* Majority attack; *PoS:* Majority attack [102]; *BFT:* Consensus 34% Attack [35]; *Vote-based:* Whitewashing [40]			Majority attack [102]	*Multiple Chains (Shards):* Majority attack [102]		*Solidity & Vyper:* Parity multi-signature wallet attack [27]
Privilege Escalation	*Public:* Majority attack [102], Sybil attack [126]					Majority attack [102]	*Multiple Chains (Shards):* Majority attack [102]		*Solidity & Vyper:* Parity Multi-signature wallet attack [27], BECToken Attack [27]; *Solidity:* Overflow/ underflow attack [137]; *Hyper-ledge Platform:* Sandboxing attacks [23]; *GOLANG:* Docker TOCTOU[4] Bug [23]

(Continued)

[4]Time of Check to Time of Use.

Table 3. Continued

Attack Category	Blockchain Architectural Dimensions								
	Access Type	**Storage and Computation**	**Consensus Mechanisms**	**Block Configuration**	**Key Management**	**Cryptographic Primitives**	**Chain Structure**	**Node Architecture**	**Smart Contracts**
Credential Access	*Private:* MITM [3, 50], MITB [52]; *Consortium:* MITM [3, 50], MITB [52]	*Centralised Off-chain:* MITM [3, 50]			*Single-key:* Brute Force attack [72], Malware attacks [40], Phishing [11, 44], MITB [52], MITM [3, 50], Replay attack [137]; *Multi-Signature:* Brute Force attack [72], Malware attacks [40]; *Threshold-Signature:* Brute Force attack [72], Malware attacks [40];	Brute force attack [72], Quantum attack [174]			
Collection & Exfiltration	*Public:* Deanonymisation attack [19]; *Private:* MITM [3, 50], MITB [52], Wormhole attacks [8]; *Consortium:* MITM [3, 50], MITB [52]	*Centralised Off-chain:* MITM [3, 50]			*Online-Wallet:* DNS hijacking [137], *Single-key:* MITB [52], MITM [3, 50], Crypto jacking [141], Malware attacks [40];*Multi-Signature:* Malware attacks [40]; *Threshold-Signature:* Malware attacks [40];	Malleability attack [72]			*Solidity & Vyper:* Short address attack [141], King of the Ether Throne [102], Dynamic libraries [102]; *Solidity:* DAO attack [72], Overflow/ underflow attack [137], GovernMental attack [13]; *EOS Platform* Fake EOS Attack [143], Random number attack [143]

(Continued)

Table 3. Continued

Attack Category	Blockchain Architectural Dimensions								
	Access Type	**Storage and Computation**	**Consensus Mechanisms**	**Block Configuration**	**Key Management**	**Cryptographic Primitives**	**Chain Structure**	**Node Architecture**	**Smart Contracts**
Impact	*Public:* Finney attack [3], Selfish mining [102], Eclipse attack [74], Routing attacks [31], Stealthier attack [149][5]; *Private:* Ransomware attacks [52, 128], Tampering [59], DDoS [72], DoS on endorsers[6]; *Consortium:* DDoS [72]	*Centralised Off-chain:* DDoS [72]; *Off-chain:* Brute force attack [72], Tampering [59]	*PoW:* Finney attack [3], Selfish mining [102], Eclipse attack [74], BGP hijacking [137] *PoS:* Nothing-at-stake [72]; *PBFT:* DDoS [72]	Consensus delay [72], Block withholding [137], Timejacking attack [137]	*Online-Wallet:* Flooding attack [72], DNS hijacking [137]	Brute force attack [72], Quantum attack [174]	Finney attack [3], *Multiple chains:* DDoS [72]	Tampering, Block withholding [137]; *Light node:* DDoS [72]	*Solidity & Vyper:* DoERS[7], HYIP Attack [27], ERC-20 Signature Replay Attack [27], Under-priced DDoS Attacks [27], BECToken Attack [27], Exploit Inconsistent behaviours of ERC-20 attacks [30]; *Solidity:* GovernMental Attacks [13]; *Hyperledge Platform:* Concurrency Attacks [171]; *GOLANG:* Key generation attack [171]; *EOS Platform:* Transaction Congestion Attack [143], Random number attack [143], DoS by draining EOS resources [94]; *External oracle:* DDoS [72], Tampering [59], Oracle Manipulation Attack [132]

[5]Also known as EREBUS attack.
[6]It might attack Hyperledger Fabric Platform.
[7]Denial of Ethereum Blockchain Remote Procedure Cal Service (DoERS).

Reconnaissance: The adversary's goal here is to aggregate sensitive information. To achieve this goal, an adversary might apply active scans to gather information by probing victim infrastructure via network traffic. This helps the attacker to accomplish further attacks, such as *deanonymisation* [19] attacks to a public blockchain, in hopes of identifying nodes' identities and grabbing useful information that should not be known to the adversary. ***Man-in-the-middle (MITM)*** attacks [3, 50] target private and consortium blockchains to violate node privacy or gather data exchanged between nodes and these blockchain networks. The adversary can also leverage *malware attacks* [40] to gather information that might help to achieve other objectives, such as gathering users' wallet information to steal their private keys. Also, there are multiple attacks belonging to this tactic target smart contracts. Attacks classified under this tactic are mostly intended to compromise pseudo-anonymity, confidentiality, or the privacy of targeted blockchain-based systems.

Resource Development: The adversary's goal in this tactic is to establish or compromise resources that can be exploited to support further malicious operations. Such resources include a large number of pseudonymous identities, or fake identities, that appear to be different nodes when, in reality, they are all under the control of a single party. Therefore, the attacker can gain influence and control a majority of nodes in the network; this action is known as a *Sybil attack* [126] and targets public blockchains. Block withholding [137] and *Finney attacks* [3] are intended to create conflicting views about a blockchain. These lead to hiding, forging, or withholding important information that must be transmitted across the network. Reputation-based attacks [40], such as *whitewashing* and *hiding block attacks*, are considered a part of this tactical approach. A malicious node can change its reputation from negative to positive by eliminating its current identity and creating a new one. Noticeably, attacks using this tactic mostly target the mining process and consensus mechanisms. As a result, this might affect block configuration, chain structure, and several aspects of smart contracts.

Persistence: This shows a tactical adversary's objective to maintain its presence in the system. This goal can be achieved when a single attacker keeps creating Sybil nodes to dominate a majority of the network's hash rate and manipulate blockchain transactions to their advantage. This is known as a *majority attack* [102] and is mostly used to target public blockchains. The adversary might also target smart contracts and exploit access control vulnerabilities to change the contract owner and control every transaction invoked in the contract. *Parity multisignature wallet attacks* [27] are a well-known type of attack that targets Ethereum smart contracts. This hack is an instance of exploiting a well-written library code once it is used in a non-intended context. The library's initialisation function could be externally called, which allows the attacker to set himself as the owner of the contract. After taking over the contract, the attacker calls the suicide method to kill the contract; this was once done, causing a permanent freeze of US$280M in the affected wallets. Attacks belonging to this tactic can also target consensus mechanisms, cryptographic primitives, and multiple chains structure architectural dimensions.

Privilege Escalation: In this sort of attack, the adversary is attempting to obtain a higher level of permission in a blockchain system. Adversaries can exploit smart contract vulnerabilities to elevate their permissions and perform unauthorised actions. *Parity multisignature wallet attacks* are one instance of an attack that falls under this category. Attacks belonging to this tactic can also target public access type, cryptographic primitives, and multiple chains structure. It is worth mentioning that this category of attack often overlaps with persistence attacks, as the exploited weaknesses that let an adversary persist can be exploited in an elevated context.

Credential Access: Attacks in this category aim to gain access to resources by exploiting the vulnerabilities within the system identification and authentication mechanisms to expose sensitive

data or transactions and/or manipulate them. Such attacks include quantum attacks [174] that can derive private keys from public keys, brute-force attacks [72], ***man in the browser (MITB)*** [52], and *malware attacks* that can steal the keys and credentials of users' online wallets. Adversaries may attempt to hijack network traffic using *MITM* techniques to collect nodes' sensitive information. Employing legitimate credentials allows adversaries to gain access to the system and makes attackers' actions harder to detect. These types of attacks can also target private and consortium blockchain and centralised off-chain storage.

Collection & Exfiltration: We combined these two adversarial tactics into one section, as adversaries often need to gather sensitive data by applying collection techniques before attempting to steal data via exfiltration techniques. Such attacks include *MITB*, where an adversary injects malware into a node's browser to collect sensitive data such as wallet credentials, which then allows the adversary to steal users' private keys. Additionally, an adversary can change the unique digital signature of a transaction before it is assigned to the chain, a process known as a *malleability attack* [72]. *DNS hijacking attacks* [137] aim to redirect users to malicious websites to collect seed phrases and private keys from users to allow the attacker to access users' wallets and steal their funds. According to Reference [32], in 2021, two cryptocurrency portals faced this type of attack. Attacks under this category mostly target users' private keys to manipulate their transaction and steal their money. Additionally, attackers can target smart contracts and identify their weaknesses to steal the cryptocurrencies stored in them. Such attacks have included *DAO attacks* [72], *GovernMental attacks* [13], and *King of the Ether Throne attacks* [102]. In each of these attacks, attackers were able to drain millions from compromised contracts. Attacks belonging to this tactic can also target access type and centralised off-chain storage.

Impact: The adversary's objective in this tactic is to damage, disrupt, or manipulate the blockchain system and its transactions. In particular, *DDoS attacks* [72] are a part of this category. The adversary uses legitimate operations to make connections, but then consumes resources to prevent other legitimate connections from requesting a particular service. Such an attack can be applied in a centralised off-chain storage system to prevent legitimate users from gaining access to stored records. Additionally, the attacker might use reconnaissance and collection techniques to hijack the connection between the external server and the blockchain system to tamper with the data. *Selfish mining attacks* [102], which target public blockchains, also fall under this category, where malicious miners collude to increase their benefits by causing honest miners to waste processing power creating blocks that will not eventually be linked to the chain. Meanwhile, the selfish miners can keep their mined blocks private, in an effort to maintain a private branch that is longer than the public branch. These selfish miners can then reveal their branch, and the honest miners will switch to it. As a result, the selfish miners win and are rewarded, while the honest miners lose and waste their power. Attacks classified in this category can be linked to all the architectural dimensions of the blockchain system. Noticeably, attacks in the impact category include only those affecting the integrity or availability of blockchain systems' information or transactions.

One insight from the table is that a notable proportion of attacks target public blockchain and the consensus mechanisms related to it. If an architect decides to design a public blockchain-based system, then they need to recognise adversaries' tactics and attack techniques that often target this type of blockchain access. As Table 3 shows, public blockchains are prone to a significant number of attacks compared with private ones. Because of the characteristics of private blockchains, several attacks—including Sybil attacks, selfish mining attacks, and majority attacks—are difficult to launch and easy to prevent. Furthermore, a PoW consensus mechanism, which is often applied in public blockchains, is prone to the last two mentioned attacks, while PoS has been proposed to mitigate the risks of these attacks. However, since the records are securely distributed in public

blockchain networks, they are more resilient against DDoS. Moreover, ransomware attacks [52] are difficult to achieve, as locking down redundant records across the whole public network is a complicated task.

Another insight is that in blockchain, a user's private keys are the most vulnerable point of attack. These keys can be compromised not only through exploiting the vulnerability in the digital signature cryptographic primitive, but also by attacking the wallets where these keys are stored. In particular, online wallets, which store the private keys in web servers, are prone to attacks that often target web applications. Moreover, several attacks that target smart contracts have been reported that have resulted in losses of millions of dollars including DAO attacks and Parity Multi-Sig Wallet attacks. To this end, architects can leverage Table 3 to recognise the adversarial tactics and attacks that target each architectural dimension when they decide to design blockchain-based systems.

4.1.2 Linking STRIDE Threats with Blockchain Systems Architectural Decisions. Exploration of the relationship between blockchain architectural decisions and their consequent threats is presented in Table 4. The explanation of each STRIDE threat category in the context of blockchain systems is as follows.

Spoofing: This threat refers to the attempt on the part of an adversary to access a blockchain system, or even control the network, by using a false identity. This can be done by stealing or retrieving the private keys and the credentials of an authorised node. Subsequently, the adversary can successfully access the victim's account or wallet to engage in illegitimate activities such as abusing transactions or violating the victim's privacy. Moreover, the adversary can create multiple fake accounts, Sybil nodes, to gain control of a consensus protocol (as explained in Section 5.1.1, Resource Development paragraph) and accomplish malicious behaviour such as double spending. When the adversaries with their malicious nodes control a majority of the network, they can alter the entries on the distributed ledger to make the payments disappear after they have been spent [121].

Tampering: In this threat, an attacker attempts to accomplish unauthorised alterations to data, transactions, or blocks that are recorded in the storage or those that are being transferred through the network. Particularly, when an online wallet is used to perform transactions, the attacker attempts to hijack the session and modify the out-going transaction to his benefit. Furthermore, attackers equipped with quantum computers will be able to apply the Shor algorithm [97], which can find the prime factorisation of large numbers and solve discrete logarithms in polynomial time. Consequently, the digital signature algorithms utilised in most current blockchain-based systems can be breached. This will allow attackers to easily derive private keys from public keys to alter transactions and sign them on behalf of the victim.

Repudiation: This is the ability of malicious participants or attackers to leverage the inability of the system or other participants to track the malicious actions or transactions that they have performed. The attacker can modify the blocks in the chain and the hashed meta-data stored on the chain once he can crack the hash function by finding hash collisions. Furthermore a quantum adversary, who can apply a quantum algorithm such as Grovers' search algorithm, can search for hash collisions significantly faster $O(\sqrt{n})$ than in the case of a classic brute force attack O(n) [97]; in the future, an attacker will also be able to replace blocks in the chain without affecting the integrity of the blockchain system.

Information Disclosure: This refers to exposing private information to individuals who are not permitted to have access to it. If the attacker successfully derives the private key from the

Table 4. Linking STRIDE with Blockchain Architectural Decisions

STRIDE Threat	Blockchain Architectural Dimensions								
	Access Type	**Storage and Computation**	**Consensus Mechanisms**	**Block Configuration**	**Key Management**	**Cryptographic Primitives**	**Chain Structure**	**Node Architecture**	**Smart Contracts**
Spoofing	Public: Double spending [121] Private/ Consortium: Privacy violation [10], Untrusted identities [10], steal-front end login information [53, 103]	Centralised off-chain storage: Gain access to the storage [168]	Voting-based: Out-vote by fake accounts [72]. PoS: Generating different blockchains with old accounts [72].		Local wallet and On-line wallet: Buggy software installation [88], Compromised private key [3], Packet spoofing [82].	Transaction spoofing [173], Recovering of the private key [102], Compromised private key [3]			Change contracts owner [75].
Tampering	Private: Transaction manipulation [140]	On-chain: modify the hashed met-data [168]. Off-chain Computation: Transaction manipulation [140]	PoW: Control transaction's confirmation [102], Acquire dominance in the pools [72], Block modification [140].	Control transaction's confirmation [102].	Impersonation at future transaction [140], Transaction manipulation. Online-Wallet: Alter out-going data [52], Alter transaction history.	Digital-signature: Shor's quantum algorithm [62], Transaction malleability, Impersonation at future transaction [140], Transaction manipulation [140]. Hash: Grover's search algorithm [97], Transaction manipulation [140], Block modification [140]	Control the confirmation operation [102], Control transaction's confirmation. Off-chain Channel: Transaction manipulation [140]	Light node: Fake header [36]	Malicious external oracle [169], Change contracts owner [75], External oracle: Unfair income [12], Critical unwanted behaviors [28], Money frozen [132]
Repudiation	Private: Transaction manipulation [140]		PBFT: Untrustworthy nodes [168]. PoW: Control the confirmation operation [102], Acquire dominance in the pool, Block modification [140].	Control the confirmation operation [102]	Impersonation at future transaction [140].	Hash: Grover's search algorithm [97], Transactions manipulation, Block modification [140]. Digital-signature: Shor's quantum algorithm [62].			Malicious external oracle [169], Untrustworthy data feeds [13], Unfair income [12].
Information Disclosure	Public: Sensitive data exposure [10], Privacy violation [10], Private/ Consortium: Steal-front end login information [103], Eavesdropping [3]; Private: leakage of confidential information [8].	On-chain: Sensitive data exposure, Privacy violation [10].			Single-key: Eavesdropping [3], Data exposure, Privacy violation; Multi-sig: Eavesdropping [3], Privacy violation; [10], On-line wallet: Bypass credential validation [24], Compromised key [63].	Compromised key, Transaction pattern exposure [58], Transaction graph analysis [58].	Single chain: Sensitive data exposure [10], Privacy violation [10]	Light node: Privacy violation [10], Untractability violation [10].	Information leakage.

(Continued)

Table 4. Continued

STRIDE Threat	Blockchain Architectural Dimensions								
	Access Type	**Storage and Computation**	**Consensus Mechanisms**	**Block Configuration**	**Key Management**	**Cryptographic Primitives**	**Chain Structure**	**Node Architecture**	**Smart Contracts**
Denial of Service	Private/ Consortium: Exhausting computational resources [102], Nodes flooding/isolation [72], Massive transaction backlogs [72].	Centralised Off-chain: Compromise the availability [169], Data loss [168].	PoW: Exhausting computational resources [102], nodes flooding/isolation, massive transaction backlogs.	Increase Block Size: Network congestion [33], Decrease transaction throughput [65].	Single-Key: Compromise the availability, Online-Wallet: Server flooding [72].		Multiple chains: Compromise the availability.	Light node: Compromise the availability.	resource-consuming procedure [99], Untrustworthy external calls [72], Untrustworthy data feeds [13], Disturbing external oracle [13], Compromise the availability, temporary shutdown of token trading [27]
Elevation of Privileges	Public: Splitting mining power [110], Engineering block races, Modifying transactions, Create blockchain forks [105], Race conditions by forking [136]. Private-Consortium: Untrusted identitiey.	Centralised Off-chain Storage: Gain access to the storage.	PoW: Double-spending, Modifying transactions, Control the confirmation operation [102]. PoS: Double-spending, Modifying transactions, Control the confirmation operation [102]. DPoS: Double-spending, Collude threats [102]	Double-spending, Blockchain forks [105], Conflicting, Stale Block [65].	Crafting malicious Payload into high privilege extension [88], Bypass credential validation [24]; Online-wallet: Crafting malicious Payload into high privilege extension [88], Bypass credential validation, Local-Wallet: Bypass credential validation.	Digital-signature: Shor's quantum algorithm [62]. Hash: Grover's search algorithm [97], Double spending [121].	Double spending [121], Blockchain forks [105].		Destroyable contract [22], Change contract owner [75], Stolen tokens [27].

user's public key, as explained in the section dealing with a tampering threat, then users will lose their privacy. In a public blockchain, if effective privacy-preserving mechanisms are not in place, then the attacker can trace transactions and eventually link the user's pseudonym to the user's real identity [19].

Denial of Service: The aim of this threat is making a system unavailable when legitimate users request a service. This can be accomplished by causing network congestion that interrupts the service available to the user. Even though a blockchain network presents resistance against this threat [72], blockchain-based systems are prone to this category of threat. This is because the node can be flooded with a large amount of junk data to exhaust its computational resources and prevent it from performing normal transactions. Additionally, blockchain systems might have a single point of failure component that is vulnerable to a denial of service threat. In particular, if smart contracts request data from a single external oracle and wait for its response to accomplish subsequent operations, then an attacker can target this server by bombarding it with requests to prevent it from responding to legitimate smart contract requests.

Elevation of Privileges: This threat occurs when malicious users with restricted privileges succeed in gaining access to a system or a network to perform unauthorised activities. For example, this can be accomplished by an attacker who can trick a user of an online-wallet to install a malicious payload into a high-privilege extension to gain access to the victim's wallet and alter the transactions. Another example is the usage of large block sizes that may cause chain forks [105], resulting from the increasing number of stale blocks [65], which leads to a significant mining power loss and limits the growth of the main chain. As a consequence, this decision allows malicious miners to elevate their privilege level through (i) colluding to compromise and control the consensuses mechanism; and (ii) performing malicious activities such as establishing their private chain to create conflicting transactions with higher chances of double spending threats.

Table 4 shows that threats exist in almost all architectural aspects of blockchain systems. In particular, private keys and their management face significant threats. One important insight is that the tampering category may potentially threaten all architectural aspects of blockchain systems. Taking sub-optimal choices when engineering blockchain system leads to threats that could potentially affect the other architectural dimensions of the system. Blockchain systems architects can use this table to better understand the potential threats that each architectural decision might face. The information in this table can therefore serve as a checklist for architects as they make or review design decisions.

4.1.3 Linking Attack Categories with STRIDE Threats. In Table 5, we have linked attack categories and STRIDE threats. One insight from the relationships shown in this table is that most categories of attacks can cause nearly all threat types. These attack categories include resource development, privilege escalation, credential access, collection & exfiltration, and impact. Moreover, some of these categories may pose a significant number of threats under each threat type such as the impact category. Another insight is that other attack categories such as reconnaissance pose a specific threat type as spoofing and information disclosure.

This classification supports the identification of potential attacks that can exploit known vulnerabilities in blockchain system components and specifies the posed threats. Vulnerability identification approaches and tools can then be used to determine the specific flaws in the chosen system components. Particularly, there is a set of static and dynamic analysis tools that can identify security vulnerabilities in smart contract components [106, 135]. The information on the identified vulnerabilities allows a determination of the attack patterns that might exploit them. Our

Table 5. Linking Attacks with Threats

Attack Tactics	STRIDE Threats					
	Spoofing	**Tampering**	**Repudiation**	**Information Disclosure**	**Denial of Service**	**Elevation of Privilege**
Reconnaissance	Privacy violation [10], Transaction spoofing [173]			Privacy violation [10], Information leakage, Sensitive data exposure [10], Transaction graph analysis [58], employment analysis, Eavesdropping [3], Untractability violation [10], Eavesdropping [3], Transaction pattern exposure [58]		
Resource Development	Out-vote by fake accounts [72], generating different blockchains with old accounts [72], Change contracts owner [75], Untrusted identities [10], Double spending [121]	Control the confirmation operation [102], Acquire dominance in the pools, Unfair income [12], Change contract owner, Alter transaction history	Control the confirmation operation, acquire dominance in the pool, Untrusted identity, Unfair income		Exhausting computational resources [102], Nodes flooding/ isolation, Data loss [168].	Double-spending, Splitting mining power [110], Engineering block races, Untrusted identity, Create blockchain fork, Race conditions by forking [136], Control the confirmation operation [102], Collude threats [102], Conflicting, Stale Block [65], Change contract owner
Persistence	Double spending [121], Out-vote by fake accounts [72], Generating different blockchains with old accounts [72], Untrusted identities [10]	Impersonation at future transaction [140], Block modification [140], Control the Confirmation Operation [102], Acquire dominance in the pools, Fake header , Unfair income [12]	Block modification [140], Control the Confirmation Operation, Acquire dominance in the pools, Untrusted identities [10], Unfair income [12]			Block modification [140], Double-spending, Untrusted identity, Control the confirmation operation [102]
Privilege Escalation	Double spending, Generating different blockchains with old accounts [72], Change contracts owner [75], Untrusted identities [10], Out-vote by fake accounts [72], Privacy violation [10]	Control the Confirmation Operation, Acquire dominance in the pools, Unfair income [12], Change contracts owner [75]	Control the Confirmation Operation [102], Acquire dominance in the pools, Untrusted identities [10], Unfair income [12]	Privacy violation [10], Sensitive data exposure [10]		Double-spending, Untrusted identity, Control the confirmation operation [102], Change contract owner, Critical unwanted behaviors [28], Stolen tokens [27]
Credential Access	Privacy violation [10], steal-front end login information [103], Gain access to the storage [168], Buggy software installation [88], Compromised private key [3], Transaction spoofing [173], Recovering of the private key [102]	Transactions Manipulation, Shor's quantum algorithm [62], Transaction malleability, Alter out-going data [52], Grover's search algorithm [97]	Transaction manipulation [140], Shor's quantum algorithm, Grover's search algorithm [97]	Privacy violation [10], Sensitive data exposure [10], Eavesdropping [3], Steal-front end login information [103], Bypass credential validation [24], Compromised private key [3]		Transaction manipulation [140], Shor's quantum algorithm [62], Grover's search algorithm [97]

(Continued)

Table 5. Continued

Attack Category	STRIDE Threats					
	Spoofing	**Tampering**	**Repudiation**	**Information Disclosure**	**Denial of Service**	**Elevation of Privilege**
Collection & Exfiltration	Privacy violation [10], Change contracts owner [75], steal-front end login information [103], Compromised private key [3], Transaction spoofing [173], Identity theft, Gain access to the storage [168]	Transaction manipulation [140], Alter out-going data [52], Fake transaction, Identity theft, Change contract owner, Malicious external oracle [169]	Transaction manipulation [140], Malicious external oracle [169]	Privacy violation [10], Transaction graph analysis [58], Sensitive data exposure [10], Eavesdropping [3], Steal-front end login information [103], Bypass credential validation [24], Compromised private key [3], Untractability violation [10], Transaction pattern exposure [58], leakage of confidential information [8]		Transaction manipulation [140], Change contract owner, Gain access to the storage [168], Bypass credential validation [24]
Impact	Double spending [121], Recovering of the private key [102]	Control the confirmation operation [102], Acquire dominance in the pools, Unfair income [12], Modify the hashed met-data [168], Shor's quantum algorithm [62], Grover's search algorithm [97], Untrustworthy data feeds [13], Malicious external oracle [169], Money frozen [132]	Control the confirmation operation [102], Acquire dominance in the pools, Unfair income [12], Grover's search algorithm [97], Shor's quantum algorithm [62], Malicious external oracle [169], Untrustworthy data feeds [13]	Compromised private key [3]	Compromise the availability [169], Exhausting computational resources [102], Nodes flooding/isolation [72], Massive transaction backlogs [72], Data loss [168], Compromise the availability [169], Network congestion [33], Disturbing external oracle [13], Data loss [168], Decrease transaction throughput [65], Server flooding [72], Blockchain ingestion, Untrustworthy data feeds [13]	Double spending [121], Splitting mining power [110] , Engineering block races, Create blockchain forks [105], Race conditions by forking [136], Control the confirmation operation [102], Collude threats [102], Stale Block [65], Destroyable contract [22], Gain access to the storage [168], Shor's quantum algorithm [62], Grover's search algorithm [97], Acquire dominance in the pools, Critical unwanted behaviors [28].

mapping approach assists an analyst in identifying the attacks and their corresponding threats in each architectural dimension of blockchain systems.

5 APPLICATION OF THE TAXONOMY AND MAPPING APPROACH

In this section, we provide a three-step decision-making process to be applied to each architectural dimension illustrated in the taxonomy:

(1) Determine the quality attributes that are provided and not provided by each design alternative (using Table 2) to recognise the quality tradeoffs and select among the alternatives with quality rationale in mind.
(2) Identify attack tactics and techniques (using Table 3) that might compromise each design alternative to understand the adversarial objectives and potential methods for attacking the system.
(3) Identify the potential threats to each design alternative (using Table 4) and the specific threats posed by each attack (using Table 5).

Systematically applying these steps supports security engineers in establishing a risk management approach and effectively prioritising and mitigating the highest security risks that threaten blockchain systems by implementing effective countermeasures.

5.1 Key Management as a Case to Demonstrate Instantiating the Taxonomy and Its Mapping to Attacks and Threats

In this section, a concrete example of an existing blockchain-based system is leveraged to show how the proposed taxonomy and guidelines can be applied in practice. This blockchain system manages and shares **electronic medical record (EMR)** data for cancer patient care. The framework was proposed by Dubovitskaya et al. [47] in collaboration with the Stony Brook University Hospital. They utilise blockchain technology to maintain immutable and verifiable records that keep track of all actions across the network. This helps to improve the integrity of sensitive medical data, reducing the time needed to share EMRs as well as the overall cost. Figure 4 shows the EMR system's architectural decisions that we were able to extract or infer from their paper. The identified decisions are presented based on our taxonomy.

The EMR blockchain system is a permissioned consortium access type as patients' data might be transferred to several hospitals. This access type safeguards the privacy of highly sensitive data about patients. A fast response time is essential in medical systems, which can be provided easily by this access type. Moreover, in a permissioned consortium blockchain, there is no need to pay for the execution of a transaction, and this increases the usability of the system.

Patient metadata is stored on-chain, while two off-chain storage locations are used to store patients' raw data: an in-hospital database that stores oncology-related data and a cloud storage database that organises patients' data and encrypts the saved data with a symmetric key for each patient. A doctor can access data in the cloud according to the permission policy specified by the patient. These two off-chain storage sites reintroduce centralisation into the blockchain system and can function as a single point of failure. The EMR blockchain system applies a PBFT algorithm, which is a vote-based consensus mechanism. This is because all users in a medical application are known (patients and doctors), and only a predefined set of nodes can participate in the consensus mechanism. This type of mechanism protects against Sybil and Majority attacks.

This medical application was built on top of the Hyperledger platform, which implements smart contracts in the form of chaincode, comprising logic and correlated state components. Chaincode is written in the Go programming language. In Hyperledger, the size of the block is often 98 MB, and the block confirmation takes one second [80]. SHA-256 is used as the default hash in the

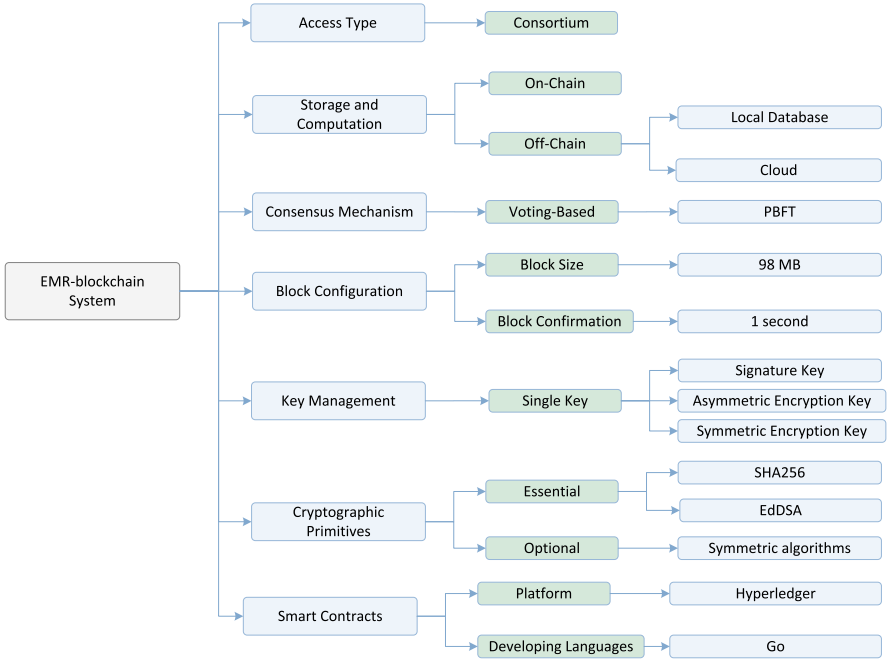


Fig. 4. Architectural decisions for EMR system.

Hyperledger platform, and the EdDSA scheme is used for digital signatures. A symmetric algorithm is used to encrypt clinical data stored in the cloud repository to provide data confidentiality.

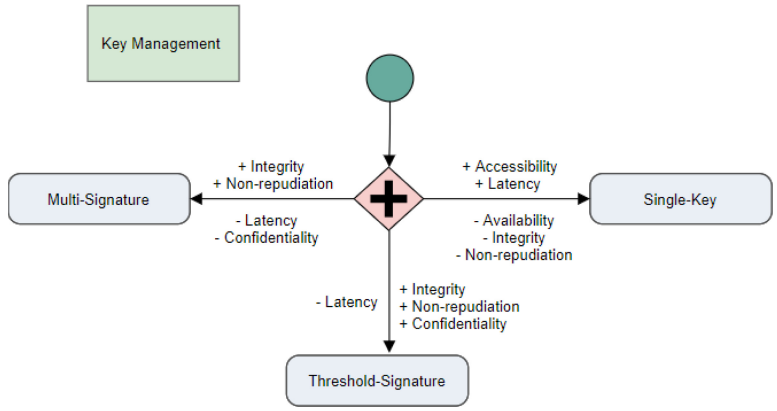
Each doctor has a public key pk_U^S and a private key sk_U^S for signing, as well as pk_U^ε and sk_U^ε for encryption. The patient can generate a metadata record on the chaincode, retrieve it, and specify permissions. In addition to the two key pairs, the patient has an asymmetric encryption key $SK^{A\varepsilon S}$ that encrypts and decrypts patient data. If a patient needs to enable a doctor to access his data, then he should encrypt the $SK^{A\varepsilon S}$ with the encryption public key of the doctor pk_D^ε, and then share the encrypted value with the doctor. Since each user has a single key pair for signing and another single key pair for encryption, these keys become a single point of failure. If attackers compromise them, then they could sign and encrypt data themselves. Moreover, if passive attackers compromise the patient's symmetric key, then they could decrypt and observe data.

We aim to investigate alternative architectural choices regarding key management dimensions, as the current choice is sub-optimal, and this might affect the security of the system. We apply our proposed three-step process to make secure and informative key management choices, as shown in Table 6 and describe bellow.

Step 1: We used **Business Process Model and Notation (BPMN)** to represent the quality attributes provided and not provided by each key management choice. The single key option provides higher accessibility and reduced system latency, but it reduces system integrity, availability, and non-repudiation. The digital signature option provides higher integrity and non-repudiation but decreases system confidentiality and leads to higher latency. Although a threshold-signature may increase system latency, it increases system confidentiality, integrity, and non-repudiation.

Step 2: We identify the tactics of potential attacks that each alternative key management design choice might be vulnerable to. Noticeably, the single key option is prone to larger sets of attacks than the other two options. Threshold signatures provide better mitigation against

Table 6. Analysis of Alternative Decisions for EMR System

Analysis of Alternative Decisions		
Quality Attributes		
Alternatives	**Potential Attacks**	**Potential Threats**
Single Key	**Reconnaissance:** Malware attacks, MITM. **Credential Access:** Brute Force, MITM, MITB, Malware attacks, Replay attacks, phishing. **Collection and Exfiltration:** MITM, MITB, Malware attacks. **Impact:** Flooding attack.	**Tampering:** Impersonation at future transaction, Transaction Manipulation. **Repudiation:** Impersonation at future transaction. **Information Disclosure:** Privacy violation. **Denial of Service:** Compromise the availability of the key, server flooding. **Elevation of Privileges:** Crafting malicious payload into high privilege extension, Bypass credential validation.
Multi-Signature	**Reconnaissance:** Malware attacks, Deanonymisation attacks. **Credential Access:** Brute Force, Malware attacks. **Collection and Exfiltration:** Malware attacks.	**Tampering:** Impersonation at future transaction, Transaction Manipulation. **Repudiation:** Impersonation at future transaction. **Information Disclosure:** Privacy violation. **Elevation of Privileges:** Crafting malicious payload into high privilege extension, Bypass credential validation.
Threshold Signature	**Reconnaissance:** Malware attacks, **Credential Access:** Brute Force, Malware attacks. **Collection and Exfiltration:** Malware attacks.	**Tampering:** Impersonation at future transaction, Transaction Manipulation. **Repudiation:** Impersonation at future transaction. **Elevation of Privileges:** Crafting malicious payload into high privilege extension, Bypass credential validation.

deanonymisation attacks than multi-signatures, as the signed transactions are indistinguishable from non-threshold transactions on the blockchain.

Step 3: We identify the potential security threats for each key management design option. The single key choice is prone to five threat categories and is the only choice that is threatened by denial of service. If this key is stored in an online wallet, then the server might receive a massive number of requests with the aim of exhausting its resources and compromising the availability of the key. Threshold signature scheme is prone to the least number of threats compared with the other two choices. Applying this scheme protects users' privacy when it's compared to multi-signature scheme where the adversary can link users' identity by tracing their multiple keys.

Based on the three-step analysis, it appears that a threshold signature would enhance the security and privacy of the EMR system. Nevertheless, security architects need to conduct a risk assessment to quantify the potential risk exposures and thus make an informed decision.

This brief example has illustrated that our approach provides a systematic way of assisting software engineers who are attempting to build a blockchain system. It also aids engineers who need to analyse and improve the security of an existing system. We have used key management as an example to demonstrate the instantiation of our taxonomy and its mapping to both threats and attacks. The same steps can be applied for analysing the attack tactics and the implied threats on any blockchain-based technique. For the case of the blockchain-based EMR system, as an example, the security of other architectural choices, as identified in Figure 4, can be analysed in the same way of that of key management. Our work provides systematic guidance for security engineers and architects, where the specific analysis techniques (e.g., prediction, estimation) for each step may vary based on the context and may be influenced by the availability of expertise in using these techniques.

6 DISCUSSION

Here, we discuss how we validate our work and the threats to validity to our approach.

6.1 Validation

A taxonomy can be validated in three ways to ensure its reliability and usefulness [150]: orthogonality demonstration, benchmarking, and utility demonstration.

6.1.1 Orthogonality Demonstration. Shows the dimensions and the categories of the taxonomy, as Figure 2 and Section 3 demonstrated. We performed an iterative content analysis method to identify the dimensions of the proposed taxonomy. We continuously evolved our taxonomy whenever a new concept was encountered in the literature. We strove to ensure the generality of each dimension by noting when several terms appeared in the literature referring to the same concern (e.g., consensus protocol vs. consensus mechanism). Moreover, a new class is introduced only when clear-cut evidence of its relevance and significance justify its inclusion. As an example, we included consensus algorithm as a class, because it has unique characteristics, properties, and well-defined terms as used in the literature. However, when there was an uncleared agreement on the term (e.g., node architecture), we came with a general class to encompass concerns and properties; nevertheless, some of the sub concerns can be further refined.

6.1.2 Benchmarking. Compares the taxonomy to related classification schemes. Only two prior works have provided taxonomies of architectural aspects of blockchain [138, 169]. However, our taxonomy is novel, as each dimension of our taxonomy has been discussed from a security perspective and mapped with the potential attacks and associated threats. Additionally, the taxonomy presented by Xu et al. [169] focuses only on six architectural components of blockchain systems, whereas the taxonomy presented here captures nine major dimensions for blockchain architectural decisions, and the discussions are explicitly focused on security. Different from Xu et al.'s taxonomy, three main architecture design dimensions—key management, cryptographic primitives, and node architecture—have been considered and thoroughly discussed in our study. Moreover, unlike theirs, our taxonomy has been derived by conducting a systematic literature review of studies related to architectural design decisions relevant to blockchain systems. Salah et al. [138] presented a taxonomy that only targeted blockchain-based artificial intelligence applications; the architectural coverage of the paper was limited to the intelligence component. Conversely, the taxonomy proposed in this work is adequate for designing blockchain-based systems in general.

6.1.3 Utility Demonstration. Is a mechanism to validate the benefits that could be gained from the taxonomy. There are several ways to demonstrate the benefits of the taxonomy as Reference [150] stated, including expert opinion and instantiation. Our taxonomy has been reviewed by an expert who gave us substantial suggestions for refinements. Additionally, the instantiation of our taxonomy was presented in Section 5.1.

6.2 Threats to Validity

Based on Reference [160], four potential threats to validity may affect our findings:

Internal Validity. One threat comes from the inherent nature of taxonomies: We cannot guarantee the completeness of our taxonomy, since there may be additional architectural design decisions that could enrich or refine the taxonomy. To mitigate this threat, we iteratively refined our taxonomy each time a new concept was encountered in the literature. Furthermore, our taxonomy is adaptable and flexible to evolve and cope with new additions and changes. Another threat comes from the possibility of considering alternative methods for threat and attack categorisation. However, the Microsoft STRIDE threat model and MITRE's attack tactics categorisation were used in our study, since they are widely used, they are consistent with current practice, and they ensure an extensive coverage of potential threats and attacks.

Construct validity. Another threat arises because our taxonomy was mainly based on the results of surveying the literature. We believe that additional sources of information could improve the completeness of this taxonomy. We mitigated this by searching the findings on multiple data sources. Additionally, our automated search was complemented by a manual search. Another threat arises from the provided set of attacks. Even though we have illustrated various kinds of attacks that pose threats to blockchain-based systems, it is impossible to cover all attacks, because new attack types are always emerging. Therefore, readers should be aware that the provided set of attacks is continually evolving, as it is difficult to predict the state of the attackers. In this study, all presented attacks are informed not only by research papers but also by technical reports, developers' blogs, and wiki pages. In any case, our proposed technique and methods for classification can be applied to categorise new and emerging attacks.

Conclusion Validity. There is a threat regarding the possibility that we interpreted the extracted data differently. The potential for bias introduced during the data extraction process was at least partially mitigated by ensuring a common understanding by all reviewers. We also ensured that the data extraction process was aligned with the research question.

External Validity. A final threat is related to the need to instantiate the taxonomy in different contexts to assist in its refinement and validation. We demonstrated the utility of the taxonomy using an example of a blockchain-based health care system. We demonstrated the applicability of our guidelines by analysing the key management architectural dimension to enhance the security of such a system. Nevertheless, further validation of the utility of the taxonomy is needed to address this threat.

7 RELATED WORK

A wide range of prior literature has discussed the properties, characteristics, and structure of blockchains. Some of them have focused on architectural components of a specific blockchain application, such as blockchain-based IoT [59], while others illustrated and discussed the security and privacy issues of blockchain technology [156]. To the best of our knowledge, no previous studies have classified the architectural design decisions of a blockchain-based system based on a systematic survey and then mapped them to threats and attacks. As shown in Table 7, we have

Table 7. Summary of Related Work

Categorisation	Ref No	Year	Contribution	Focusing Area
Blockchain Architecture	[167]	2016	Explored blockchain from an architecture point of view. Compared blockchain with other software solutions	General Blockchain-Based System
	[169]	2017	Proposed taxonomy of some of the architectural components of blockchain systems. Showed how different architectural decisions affect the quality attributes of blockchain systems	General Blockchain-Based System
	[180]	2017	Reviewed the properties of blockchain systems. Mainly investigated, compared and analysed different consensus mechanisms	General Blockchain-Based System
	[61]	2018	Illustrated the blockchain frameworks. Reviewed some blockchain applications in details	General Blockchain-Based System
	[138]	2019	Provided a detailed survey on blockchain, platforms, consensus protocols and applications that are adequate for AI area	Artificial Intelligence
	[109]	2019	Reviewed some of the blockchain components. Provided a a comprehensive discussion of the main properties of the state-of-the-art blockchain applications	Blockchain Application
Security and Privacy Issues	[102]	2017	Illustrated security risks and attacks over blockchains systems. Demonstrated several solutions protocols	General Blockchain-Based Systems
	[72]	2019	Extensively investigated vulnerabilities in each blockchain generation. Explained potential attacks. Highlighted possible countermeasures	General Blockchain-Based Systems
	[176]	2019	Explained the required security properties of blockchain-based cryptocurrency. Reviewed the existing techniques to achieve security and privacy for such systems	General Blockchain-Based Systems
	[156]	2019	Reviewed the security aspects and cyberattacks of each layer of blockchain-based systems. Summarised the existing mitigation techniques	General Blockchain-Based Systems
	[137]	2020	Investigated attack surface in multiple implementations of blockchains. Outline multiple defence techniques.	General Blockchain-Based Systems
	[33]	2018	Reviewed vulnerabilities in Bitcoin and related threats. Investigated the effectiveness of the proposed solutions. Reviewed privacy threats to Bitcoin's users. Analysed the existing privacy-preserving solutions	Bitcoin
	[59]	2018	Classified the potential attacks against blockchain-based IoT applications. Provided mechanisms to enhance their security and privacy	Internet-of-Things
	[5]	2019	Overviewed the main blockchain architecture components and characteristics from IoT perspective. Discussed how blockchain properties enhance IoT security	Internet-of-Things
	[20]	2017	Provided a taxonomy of the major security vulnerabilities in Ethereum smart contracts. Illustrated the significant attacks in Ethereum smart contracts	Smart Contracts
	[116]	2018	Provided a comprehensive classification of known security vulnerabilities in Ethereum smart contracts	Smart Contracts
	[111]	2018	Reviewed security and privacy issues related to smart contracts applications	Smart Contracts
	[27]	2020	Provided a comprehensive list of known security vulnerabilities, attacks, and defences in Ethereum smart contracts.	Smart Contracts

classified the prior literature on blockchain into two major categories: blockchain architecture and security and privacy issues.

Regarding the first category, authors in Reference [167] discussed several blockchain architectural design choices and compared decentralised blockchain with other software solutions. In Reference [169], the authors presented a taxonomy of the architectural properties of blockchain-based systems and showed the impact of these properties on the performance and other quality attributes of the system. Yet, their impact on the security properties and their consequential security risks has not been covered. Moreover, a systematic review of the literature has not been conducted to represent the taxonomy. Similarly, in Reference [180], the authors briefly discussed the types of blockchain and then discussed and compared different types of consensus protocols. Also, they provided a classification of different types of blockchain applications. A detailed dissection of blockchain applications and their properties, architecture, and issues was presented in Reference [61]. Another review of blockchain applications was conducted by Reference [109]. This study reviewed blockchain and its key components and comprehensively detailed application examples of blockchain-based IoT, security, and data management. In Reference [138], a survey on blockchain technology for **Artificial Intelligence (AI)** was conducted. This study provided a taxonomy of blockchain characteristics that can be leveraged by AI applications. It summarised existing blockchain platforms and protocols that could be adopted for AI applications.

Moving to the second category, security and privacy issues, researchers investigated this area and emphasised that blockchain is not completely secure and is prone to various vulnerabilities and security risks. One pioneering article in this category was Reference [102], where the authors reviewed the security threats to blockchain and the corresponding attacks and suggested some security solutions. Similarly, the authors in Reference [72] classified security vulnerabilities based on blockchain accessibility. It also provided a detailed explanation of known vulnerabilities and subsequent potential attacks. Countermeasure techniques to improve the security of blockchain systems were also surveyed in this study. Investigation of security issues of blockchain was reviewed from other perspectives in Reference [156], which analysed the security issues of each layer of blockchain: application, smart contracts, incentive, consensus, network, and data layer. Another study [176] targeted security and privacy issues raised in blockchain-based cryptocurrency applications, highlighted security, and privacy properties required in many blockchain applications and then reviewed the techniques and mechanisms to achieve them. In Reference [137], authors investigated the attack surface in multiple implementations of blockchains in terms of cryptographic constructions, distributed system architecture, and applications. They also summarised defence measures carried out by blockchain technologies, or recommended by researchers, to mitigate the impacts of these attacks. Several attacks that target Bitcoin and its underlying protocols, such as **Proof-of-Work (PoW)**, were tabulated and analysed in Reference [33]. Their survey also investigated the effectiveness of existing solutions. Moreover, it discussed privacy-related threats and the privacy-perceiving techniques against them.

A considerable number of surveys have reviewed the incorporation of blockchain in IoT systems. Three studies, [5, 59, 73], are related to our work; they reviewed architectural components and implementation of blockchain based-IoT and the related security issues regarding such integration. In Reference [59], a detailed overview of blockchain protocols for IoT applications was provided. The authors classified and discussed the potential attacks against IoT applications implemented on a blockchain foundation. Likewise, a comprehensive survey regarding developing blockchain-based platforms, applications, and services, which were adequate for IoT applications, was carried out in Reference [5].

Since the emergence of smart contracts in blockchains the number of vulnerabilities and potential threats caused by wrong design and coding of smart contracts has increased. One review

[20] aggregated the known vulnerabilities in Ethereum smart contracts and classified them into three categories based on the level: solidity level, **Ethereum Virtual Machine (EVM)** level, and blockchain level. Similarly, authors in Reference [116] provided the same classification of vulnerabilities in Ethereum smart contracts; however, they provided a more comprehensive list. Authors in Reference [27] utilised the same classification to classify 40 vulnerabilities and analysed their root causes. This study also discussed attacks and multiple defence techniques. Another study [111] provided an overview of the privacy and security issues that can arise in different smart contract applications. Noticeably, all presented blockchain surveys have discussed its architecture and security concerns from different perspectives. However, no one provided an in-depth survey of blockchain architectural design decisions and linked them to classified threats and potential attacks that can breach the system if ill-informed design decisions are taken.

8 FUTURE DIRECTIONS AND CONCLUSION

8.1 Future Research Directions

We have identified several research areas that require more investigation.

Architectural Dimension Limitations. There is a lack of significant research on aspects of node architecture, as described in the taxonomy. Specifically, there is still a need to investigate and design more effective architectural solutions that allow nodes to receive and store transactional data more efficiently and securely. Another architectural dimension that requires more research is block configuration. Not enough research has investigated the limitations and security challenges related to block size and its propagation and how these decisions would affect the latency and the throughput of the blockchain network. The throughput of the blockchain network needs to be improved to be applicable in current production environments. At present, in most blockchain platforms, a transaction takes minutes to be confirmed. Thus, improving the confirmation latency to seconds, while still preserving security, is a key challenge.

Security Limitations. Even though blockchain has been considered as a solution to tackle DDoS attacks because of its decentralisation and distributed properties, it can be seen from our mapping that several blockchain system architectural dimensions are still vulnerable to DDoS attacks. Introducing centralised components, which become a single point of failure, into the blockchain system, makes the system highly prone to DDoS attacks. These attacks prevent the system from delivering the required services to the users. Therefore, techniques that can improve security against this type of attack need to be explored.

Public-key algorithms are prone to quantum attacks that might easily break transaction signatures. Few studies have investigated this situation and suggested alternative anti-quantum algorithms. However, there is a lack of studies that have analysed the alternative solutions as a means of enhancing the cryptographic algorithms applied in blockchain and selecting the optimal one in terms of the security attributes required by the application domain.

Methodological Limitations. Most studies have focused on proposing blockchain applications and use cases. However, there is a dearth of academic studies with regards to conducting systematic approaches and the use of decision models to assist decision-makers and architects in choosing appropriate components, patterns, and features when constructing blockchain systems. Moreover, analysing the security risks encountered during blockchain systems' development, and how they influence the outcomes, has not been investigated in the literature. Programming flexibility of smart contracts opens opportunities for attackers to compromise them. Thus, there is a crucial need for best practices, standards, and frameworks to assess the security risks in smart contracts, as attacks against such contracts are unavoidable.

8.2 Conclusion

In this article, we surveyed architectural properties and aspects of blockchain-based systems and provided a taxonomy that captures their major architectural design decisions. The taxonomy illustrates nine dimensions of architectural decisions related to access type, data storage and transaction computation, consensus mechanism, block configuration, key management, cryptographic primitive, chain structure, node architecture, and smart contracts. We provided a mapping that links attacks and the posed threats to the architectural decisions in our taxonomy. We systematically classified the attacks in blockchain systems following MITRE's attack tactics categories and then associated the attacks to their posed threats using the STRIDE threat model.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Giannis Tziakouris, former Digital Crime Analyst at Interpol, for his collaboration towards reviewing the taxonomy and the classification of attacks and threats.

REFERENCES

[1] Ryosuke Abe, Shigeya Suzuki, and Jun Murai. 2018. Mitigating Bitcoin node storage size by DHT. In *Proceedings of the Asian Internet Engineering Conference*. ACM, New York, NY, 17–23.

[2] S. Ahmadjee, C. Mera-Gomez, and R. Bahsoon. 2021. Assessing smart contracts security technical debts. In *Proceedings of the IEEE/ACM International Conference on Technical Debt (TechDebt)*. IEEE Computer Society, 6–15. DOI: https://doi.org/10.1109/TechDebt52882.2021.00010

[3] Nurzhan Zhumabekuly Aitzhan and Davor Svetinovic. 2016. Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams. *IEEE Trans. Depend. Sec. Comput.* 15, 5 (2016), 840–852.

[4] Muneeb Ali, Jude Nelson, Ryan Shea, and Michael J. Freedman. 2016. Blockstack: A global naming and storage system secured by blockchains. In *Proceedings of the Annual Technical Conference*. USENIX, 181–194.

[5] Muhammad Salek Ali, Massimo Vecchio, Miguel Pincheira, Koustabh Dolui, Fabio Antonelli, and Mubashir Husain Rehmani. 2018. Applications of blockchains in the internet of things: A comprehensive survey. *IEEE Commun. Surv. Tutor.* 21, 2 (2018), 1676–1717.

[6] Mohamed Almorsy, John Grundy, and Amani S. Ibrahim. 2013. Automated software architecture security risk analysis using formalized signatures. In *Proceedings of the 35th International Conference on Software Engineering (ICSE)*. IEEE, 662–671.

[7] American Council for Technology-Industry Advisory Council. 2021. Blockchain Playbook Online - beta. https://rb.gy/g5ci8eo. [Online; accessed 19-July-2021].

[8] Nitish Andola, Manas Gogoi, S. Venkatesan, Shekhar Verma, et al. 2019. Vulnerabilities on hyperledger fabric. *Pervas. Mob. Comput.* 59 (2019), 101050.

[9] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, et al. 2018. Hyperledger fabric: A distributed operating system for permissioned blockchains. In *Proceedings of the 13th EuroSys Conference*. ACM, New York, NY.

[10] Elli Androulaki, Ghassan O. Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. 2013. Evaluating user privacy in Bitcoin. In *Proceedings of the International Conference on Financial Cryptography and Data Security*. Springer, Berlin, 34–51.

[11] A. A. Andryukhin. 2019. Phishing attacks and preventions in blockchain based projects. In *Proceedings of the International Conference on Engineering Technologies and Computer Science (EnT)*. IEEE, Moscow, Russia, 15–19.

[12] Nicola Atzei, Massimo Bartoletti, and Tiziana Cimoli. 2016. A survey of attacks on Ethereum smart contracts. *IACR Cryptology ePrint Archive* 2016 (2016), 1007.

[13] Nicola Atzei, Massimo Bartoletti, and Tiziana Cimoli. 2017. A survey of attacks on Ethereum smart contracts. In *Principles of Security and Trust*. Springer, Berlin, 164–186.

[14] Len Bass, Paul Clements, and Rick Kazman. 2003. *Software Architecture in Practice*. Addison-Wesley Professional.

[15] Juan Benet. 2014. IPFS-content addressed, versioned, P2P file system. *arXiv preprint arXiv:1407.3561* 92 (2014), 399–406.

[16] Iddo Bentov, Ariel Gabizon, and Alex Mizrahi. 2016. Cryptocurrencies without proof of work. In *Proceedings of the International Conference on Financial Cryptography and Data Security*. Springer, Berlin, 142–157.

[17] Iddo Bentov, Charles Lee, Alex Mizrahi, and Meni Rosenfeld. 2014. Proof of activity: Extending Bitcoin's proof of work via proof of stake. *IACR Cryptol. ePrint Arch.* 2014 (2014), 452.

[18] Alysson Bessani, João Sousa, and Eduardo E. P. Alchieri. 2014. State machine replication for the masses with BFT-SMaRt. In *Proceedings of the 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. IEEE, 355–362.

[19] Alex Biryukov, Dmitry Khovratovich, and Ivan Pustogarov. 2014. Deanonymisation of clients in Bitcoin P2P network. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. ACM, New York, NY, 15–29.

[20] Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A. Kroll, and Edward W. Felten. 2015. SoK: Research perspectives and challenges for Bitcoin and cryptocurrencies. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*. IEEE, 104–121.

[21] Sarah Bouraga. 2021. A taxonomy of blockchain consensus protocols: A survey and classification framework. *Exp. Syst. Applic.* 168 (2021), 114384.

[22] Lexi Brent, Anton Jurisevic, Michael Kong, Eric Liu, Francois Gauthier, Vincent Gramoli, Ralph Holz, and Bernhard Scholz. 2018. Vandal: A scalable security analysis framework for smart contracts. *arXiv preprint arXiv:1809.03981* (2018).

[23] Sotirios Brotsis, Nicholas Kolokotronis, Konstantinos Limniotis, Gueltoum Bendiab, and Stavros Shiaeles. 2020. On the security and privacy of hyperledger fabric: Challenges and open issues. In *Proceedings of the IEEE World Congress on Services (SERVICES)*. IEEE, 197–204. DOI: https://doi.org/10.1109/SERVICES48979.2020.00049

[24] Thanh Bui, Siddharth Prakash Rao, Markku Antikainen, and Tuomas Aura. 2019. Pitfalls of open architecture: How friends can exploit your cryptocurrency wallet. In *Proceedings of the 12th European Workshop on Systems Security*. ACM, New York, NY.

[25] Fran Casino, Thomas K. Dasaklis, and Constantinos Patsakis. 2018. A systematic literature review of blockchain-based applications: Current status, classification and open issues. *Telem. Inform.* 36 (2018).

[26] Miguel Castro, Barbara Liskov, et al. 1999. Practical byzantine fault tolerance. In *Proceedings of the Symposium on Operating System Design & Implementation*. USENIX.

[27] Huashan Chen, Marcus Pendleton, Laurent Njilla, and Shouhuai Xu. 2020. A survey on Ethereum systems security: Vulnerabilities, attacks, and defenses. *ACM Comput. Surv.* 53, 3 (2020), 1–43.

[28] Jiachi Chen, Xin Xia, David Lo, John Grundy, Xiapu Luo, and Ting Chen. 2021. DEFECTCHECKER: Automated smart contract defect detection by analyzing EVM bytecode. *IEEE Trans. Softw. Eng.* (2021), 1–1. DOI: https://doi.org/10.1109/TSE.2021.3054928

[29] Lin Chen, Lei Xu, Nolan Shah, Zhimin Gao, Yang Lu, and Weidong Shi. 2017. On security analysis of proof-of-elapsed-time (POET). In *Proceedings of the International Symposium on Stabilization, Safety, and Security of Distributed Systems*. Springer, Cham, 282–297.

[30] Ting Chen, Yufei Zhang, Zihao Li, Xiapu Luo, Ting Wang, Rong Cao, Xiuzhuo Xiao, and Xiaosong Zhang. 2019. TokenScope: Automatically detecting inconsistent behaviors of cryptocurrency tokens in Ethereum. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS'19)*. Association for Computing Machinery, New York, NY, 1503–1520. DOI: https://doi.org/10.1145/3319535.3345664

[31] Jieren Cheng, Luyi Xie, Xiangyan Tang, Naixue Xiong, and Boyi Liu. 2021. A survey of security threats and defense on Blockchain. *Multim. Tools Applic.* 80, 20 (2021), 30623–30652.

[32] Catalin Cimpanu. 2021. DNS hijacks at two cryptocurrency sites. Retrieved from: https://bit.ly/3gGtO33. (2021).

[33] Mauro Conti, E. Sandeep Kumar, Chhagan Lal, and Sushmita Ruj. 2018. A survey on security and privacy issues of Bitcoin. *IEEE Commun. Surv. Tutor.* 20, 4 (2018), 3416–3452.

[34] Corda. 2019. An Open Source Blockchain Platform for Businesses | Corda. Retrieved from: https://www.corda.net/. (2019).

[35] CSA. 2020. Over 200 Documented Blockchain Attacks, Vulnerabilities and Weaknesses. Retrieved from: https://bit.ly/3kuMgwx. (2020).

[36] Leonardo da Costa, André Neto, Billy Pinheiro, Weverton Cordeiro, Roberto Araújo, and Antônio Abelém. 2019. Securing light clients in blockchain with DLCP. *Int. J. Netw. Manag.* 29, 3 (2019), e2055.

[37] Hung Dang, Tien Tuan Anh Dinh, Dumitrel Loghin, Ee-Chien Chang, Qian Lin, and Beng Chin Ooi. 2019. Towards scaling blockchain systems via sharding. In *Proceedings of the International Conference on Management of Data*. Association for Computing Machinery, New York, NY, 123–140.

[38] Florian Daniel and Luca Guida. 2019. A service-oriented perspective on blockchain smart contracts. *IEEE Internet Comput.* 23, 1 (2019), 46–53.

[39] Chris Dannen. 2017. *Introducing Ethereum and Solidity*. Springer.

[40] Dipankar Dasgupta, John M. Shrein, and Kishor Datta Gupta. 2019. A survey of blockchain from security perspective. *J. Bank. Finan. Technol.* 3, 1 (2019), 1–17.

[41] Christian Decker and Roger Wattenhofer. 2013. Information propagation in the Bitcoin network. In *Proceedings of the IEEE International Conference on Peer-to-Peer Computing*. IEEE, 1–10.

[42] Harsh Desai, Murat Kantarcioglu, and Lalana Kagal. 2019. A hybrid blockchain architecture for privacy-enabled and accountable auctions. In *Proceedings of the IEEE International Conference on Blockchain (Blockchain)*. IEEE, 34–43.
[43] Giuseppe Destefanis, Michele Marchesi, Marco Ortu, Roberto Tonelli, Andrea Bracciali, and Robert Hierons. 2018. Smart contracts vulnerabilities: A call for blockchain software engineering? In *Proceedings of the International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*. IEEE, 19–25.
[44] Digitalshadows. 2021. Cryptocurrency Attacks to Be Aware of in 2021. Retrieved from: https://bit.ly/2WyC3XG. (2021).
[45] Sheng Ding, Jin Cao, Chen Li, Kai Fan, and Hui Li. 2019. A novel attribute-based access control scheme using blockchain for IoT. *IEEE Access* 7 (2019), 38431–38441.
[46] Ali Dorri, Marco Steger, Salil S. Kanhere, and Raja Jurdak. 2017. Blockchain: A distributed solution to automotive security and privacy. *IEEE Commun. Mag.* 55, 12 (2017), 119–125.
[47] Alevtina Dubovitskaya, Zhigang Xu, Samuel Ryu, Michael Schumacher, and Fusheng Wang. 2017. Secure and trustable electronic medical records sharing using blockchain. In *AMIA Annual Symposium Proceedings*, Vol. 2017. American Medical Informatics Association.
[48] Stefan Dziembowski, Sebastian Faust, Vladimir Kolmogorov, and Krzysztof Pietrzak. 2015. Proofs of space. In *Proceedings of the Annual Cryptology Conference*. Springer, 585–605.
[49] Jacob Eberhardt and Stefan Tai. 2017. On or off the blockchain? Insights on off-chaining computation and data. In *Proceedings of the European Conference on Service-oriented and Cloud Computing*. Springer, Cham, 3–15.
[50] Parinya Ekparinya, Vincent Gramoli, and Guillaume Jourjon. 2018. Impact of man-in-the-middle attacks on Ethereum. In *Proceedings of the IEEE 37th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 11–20. DOI: https://doi.org/10.1109/SRDS.2018.00012
[51] Steve Ellis, Ari Juels, and Sergey Nazarov. 2018. Chainlink: A decentralized oracle network. https://research.chain.link/whitepaper-v1.pdf. [Online; accessed 19-August-2021].
[52] E. English, A. D. Kim, and M. Nonaka. 2018. Advancing Blockchain Cybersecurity: Technical and Policy Considerations for the Financial Services Industry. https://rb.gy/jv8rra. [Online; accessed 12-Nov-2019].
[53] Shayan Eskandari, Andreas Leoutsarakos, Troy Mursch, and Jeremy Clark. 2018. A first look at browser-based cryptojacking. In *Proceedings of the IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 58–66.
[54] Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert Van Renesse. 2016. Bitcoin-NG: A scalable blockchain protocol. In *Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI'16)*. USENIX, 45–59.
[55] Weidong Fang, Wei Chen, Wuxiong Zhang, Jun Pei, Weiwei Gao, and Guohui Wang. 2020. Digital signature scheme for information non-repudiation in blockchain: A state of the art review. *EURASIP J.ournal on Wirel. Commun. Netw.* 2020, 1 (2020), 1–15.
[56] Kurt Fanning and David P. Centers. 2016. Blockchain and its coming impact on financial services. *J. Corpor. Account. Finan.* 27, 5 (2016), 53–57.
[57] Siamak Farshidi, Slinger Jansen, Sergio España, and Jacco Verkleij. 2020. Decision support for blockchain platform selection: Three industry case studies. *IEEE Trans. Eng. Manag.* 67, 4 (2020), 1109–1128. DOI: https://doi.org/10.1109/TEM.2019.2956897
[58] Qi Feng, Debiao He, Sherali Zeadally, Muhammad Khurram Khan, and Neeraj Kumar. 2018. A survey on privacy protection in blockchain system. *J. Netw. Comput. Applic.* 126, 2 (2018).
[59] Mohamed Amine Ferrag, Makhlouf Derdour, Mithun Mukherjee, Abdelouahid Derhab, Leandros Maglaras, and Helge Janicke. 2018. Blockchain technologies for the internet of things: Research issues and challenges. *IEEE Internet Things J.* 6, 2 (2018), 2188–2204.
[60] Martin Florian, Sebastian Henningsen, Sophie Beaucamp, and Björn Scheuermann. 2019. Erasing data from blockchain nodes. In *Proceedings of the IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 367–376.
[61] Weichao Gao, William G. Hatcher, and Wei Yu. 2018. A survey of blockchain: Techniques, applications, and challenges. In *Proceedings of the 27th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 1–11.
[62] Yu-Long Gao, Xiu-Bo Chen, Yu-Ling Chen, Ying Sun, Xin-Xin Niu, and Yi-Xian Yang. 2018. A secure cryptocurrency scheme based on post-quantum blockchain. *IEEE Access* 6 (2018), 27205–27213.
[63] Rosario Gennaro, Steven Goldfeder, and Arvind Narayanan. 2016. Threshold-optimal DSA/ECDSA signatures and an application to Bitcoin wallet security. In *Proceedings of the International Conference on Applied Cryptography and Network Security*. Springer, 156–174.
[64] Craig Gentry and Dan Boneh. 2009. *A Fully Homomorphic Encryption Scheme*. Vol. 20. Stanford University, Stanford, CA.

[65] Arthur Gervais, Ghassan O. Karame, Karl Wüst, Vasileios Glykantzis, Hubert Ritzdorf, and Srdjan Capkun. 2016. On the security and performance of proof of work blockchains. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. ACM, New York, NY, 3–16.
[66] Bill Gleim. 2017. General Philosophy - Ethereum Smart Contract Best Practices. Retrieved from: https://bit.ly/3DpudAy. (2017).
[67] Steven Goldfeder, Joseph Bonneau, J. A. Kroll, and E. W. Felten. 2014. Securing bitcoin wallets via threshold signatures. https://tinyurl.com/k2cj4ee2.
[68] Steven Goldfeder, Rosario Gennaro, Harry Kalodner, Joseph Bonneau, Joshua A. Kroll, Edward W. Felten, and Arvind Narayanan. 2015. Securing Bitcoin wallets via a new DSA/ECDSA threshold signature scheme. https://tinyurl.com/3p2p2s85.
[69] Matthew Green and Ian Miers. 2017. Bolt: Anonymous payment channels for decentralized currencies. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. 473–489.
[70] Jens Groth. 2010. Short pairing-based non-interactive zero-knowledge arguments. In *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 321–340.
[71] John H. Hartman, Ian Murdock, and Tammo Spalink. 1999. The swarm scalable storage system. In *Proceedings of the 19th IEEE International Conference on Distributed Computing Systems*. IEEE, 74–81.
[72] Huru Hasanova, Ui-Jun Baek, Mu-Gon Shin, Kyunghee Cho, and Myung-Sup Kim. 2019. A survey on blockchain cybersecurity vulnerabilities and possible countermeasures. *Int. J. Netw. Manag.* 29, 2 (2019), e2060.
[73] Muneeb Ul Hassan, Mubashir Husain Rehmani, and Jinjun Chen. 2019. Privacy preservation in blockchain based IoT systems: Integration issues, prospects, challenges, and future research directions. *Fut. Gen. Comput. Syst.* 97 (2019), 512–529.
[74] Ethan Heilman, Alison Kendler, Aviv Zohar, and Sharon Goldberg. 2015. Eclipse attacks on Bitcoin's peer-to-peer network. In *Proceedings of the 24th USENIX Security Symposium (USENIX Security'15)*. USENIX Association, 129–144. Retrieved from: https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/heilman.
[75] Pete Humiston. 2018. Smart Contract Attacks [Part 2] - Ponzi Games Gone Wrong. Retrieved from: https://bit.ly/2WqVcew. (2018).
[76] Clemente Izurieta, Kali Kimball, David Rice, and Tessa Valentien. 2018. A position study to investigate technical debt associated with security weaknesses. In *Proceedings of the IEEE/ACM International Conference on Technical Debt (TechDebt)*. IEEE, 138–142.
[77] Clemente Izurieta and Mary Prouty. 2019. Leveraging secdevops to tackle the technical debt associated with cybersecurity attack tactics. In *Proceedings of the IEEE/ACM International Conference on Technical Debt (TechDebt)*. IEEE, 33–37.
[78] Hai Jin, Xiaohai Dai, and Jiang Xiao. 2018. Towards a novel architecture for enabling interoperability amongst multiple blockchains. In *Proceedings of the IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 1203–1211.
[79] Aram Jivanyan. 2019. Lelantus: Towards confidentiality and anonymity of blockchain transactions from standard assumptions. *IACR Cryptol. ePrint Arch.* 2019 (2019), 373.
[80] Aditya Joshi. 2020. Modifying the Batch Size in Hyperledger Fabric v2.2. Retrieved from: https://bit.ly/3ve5zyM. (2020).
[81] Maxim Jourenko, Kanta Kurazumi, Mario Larangeira, and Keisuke Tanaka. 2019. SoK: A taxonomy for layer-2 scalability related protocols for cryptocurrencies. *IACR Cryptol. ePrint Arch.* 2019 (2019), 352.
[82] J. W. Weatherman. 2018. Bitcoin security threat model. Retrieved from: https://bit.ly/2Y0xJkx. (2018).
[83] Satyanarayan Kar, Vinay Kasimsetty, Susan Barlow, and Sujay Rao. 2019. *Risk Analysis of Blockchain Application for Aerospace Records Management*. Technical Report. SAE Technical Paper.
[84] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. 2017. Ouroboros: A provably secure proof-of-stake blockchain protocol. In *Proceedings of the Annual International Cryptology Conference*. Springer, 357–388.
[85] Evgeniy O. Kiktenko, Nikolay O. Pozhar, Maxim N. Anufriev, Anton S. Trushechkin, Ruslan R. Yunusov, Yuri V. Kurochkin, A. I. Lvovsky, and A. K. Fedorov. 2018. Quantum-secured blockchain. *Quant. Sci. Technol.* 3, 3 (2018), 035004.
[86] Barbara Kitchenham. 2004. Procedures for performing systematic reviews. Keele, UK, Keele University 33, 2004 (2004), 1–26. https://rb.gy/qeroiv.
[87] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. 2016. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *Proceedings of the IEEE Symposium on Security and Privacy*. IEEE, 839–858.
[88] Nir Kshetri. 2017. Can blockchain strengthen the internet of things? *IT Profess.* 19, 4 (2017), 68–72.

[89] Alexandr Kuznetsov, Kyryl Shekhanin, Andrii Kolhatin, Diana Kovalchuk, Vitalina Babenko, and Iryna Perevozova. 2019. Performance of hash algorithms on gpus for use in blockchain. In *Proceedings of the IEEE International Conference on Advanced Trends in Information Theory (ATIT)*. IEEE, 166–170.
[90] Roy Lai and David LEE Kuo Chuen. 2018. Blockchain–from public to private. In *Handbook of Blockchain, Digital Finance, and Inclusion, Volume 2*. Elsevier, 145–177.
[91] Leslie Lamport et al. 2001. Paxos made simple. *ACM Sigact News* 32, 4 (2001), 18–25.
[92] David LeBlanc and Michael Howard. 2002. *Writing Secure Code*. Pearson Education.
[93] Ledger. 2020. E-commerce and marketing data breach. Retrieved from: https://bit.ly/2WI33EC. (2020).
[94] Sangsup Lee, Daejun Kim, Dongkwan Kim, Sooel Son, and Yongdae Kim. 2019. Who spent my EOS? on the (in) security of resource management of EOS.IO. In *Proceedings of the 13th USENIX Workshop on Offensive Technologies (WOOT'19)*.
[95] Kaijun Leng, Ya Bi, Linbo Jing, Han-Chi Fu, and Inneke Van Nieuwenhuyse. 2018. Research on agricultural supply chain system with double chain architecture based on blockchain technology. *Fut. Gen. Comput. Syst.* 86 (2018), 641–649.
[96] Yoad Lewenberg, Yonatan Sompolinsky, and Aviv Zohar. 2015. Inclusive block chain protocols. In *Proceedings of the International Conference on Financial Cryptography and Data Security*. Springer, 528–547.
[97] Chao-Yang Li, Xiu-Bo Chen, Yu-Ling Chen, Yan-Yan Hou, and Jian Li. 2018. A new lattice-based signature scheme in post-quantum blockchain network. *IEEE Access* 7 (2018), 2026–2033.
[98] Daming Li, Zhiming Cai, Lianbing Deng, Xiang Yao, and Harry Haoxiang Wang. 2019. Information security model of block chain based on intrusion sensing in the IoT environment. *Clust. Comput.* 22, 1 (2019), 451–468.
[99] Kai Li, Jiaqi Chen, X. Liu, Y. Tang, Xiaofeng Wang, and Xiapu Luo. 2021. As strong as its weakest link: How to break blockchain DApps at RPC service. In *Proceedings of the Network and Distributed System Security Symposium*. NDSS.
[100] Wenting Li, Sébastien Andreina, Jens-Matthias Bohli, and Ghassan Karame. 2017. Securing proof-of-stake blockchain protocols. In *Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Springer, 297–315.
[101] Wenting Li, Alessandro Sforzin, Sergey Fedorov, and Ghassan O. Karame. 2017. Towards scalable and private industrial blockchains. In *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*. ACM, 9–14.
[102] Xiaoqi Li, Peng Jiang, Ting Chen, Xiapu Luo, and Qiaoyan Wen. 2017. A survey on the security of blockchain systems. *Fut. Gen. Comput. Syst.* 107, 4 (2017).
[103] Chao Lin, Debiao He, Xinyi Huang, Kim-Kwang Raymond Choo, and Athanasios V. Vasilakos. 2018. BSeIn: A blockchain-based secure mutual authentication with fine-grained access control system for industry 4.0. *J. Netw. Comput. Applic.* 116 (2018), 42–52.
[104] Fei Lin and Minqian Qiang. 2018. The challenges of existence, status, and value for improving blockchain. *IEEE Access* 7 (2018), 7747–7758.
[105] Iuon-Chang Lin and Tzu-Chun Liao. 2017. A survey of blockchain security issues and challenges. *IJ Netw. Secur.* 19, 5 (2017), 653–659.
[106] Jing Liu and Zhentian Liu. 2019. A survey on security verification of blockchain smart contracts. *IEEE Access* 7, 2 (2019).
[107] Yue Liu, Qinghua Lu, Xiwei Xu, Liming Zhu, and Haonan Yao. 2018. Applying design patterns in smart contracts. In *Proceedings of the International Conference on Blockchain*. Springer, 92–106.
[108] Sin Kuang Lo, Xiwei Xu, Mark Staples, and Lina Yao. 2020. Reliability analysis for blockchain oracles. *Comput. Electric. Eng.* 83 (2020), 106582.
[109] Yang Lu. 2019. The blockchain: State-of-the-art and research challenges. *J. Industr. Inf. Integ.* 15 (2019). DOI: https://doi.org/10.1016/j.jii.2019.04.002
[110] Loi Luu, Ratul Saha, Inian Parameshwaran, Prateek Saxena, and Aquinas Hobor. 2015. On power splitting games in distributed computation: The case of Bitcoin pooled mining. In *Proceedings of the IEEE 28th Computer Security Foundations Symposium*. IEEE, 397–411.
[111] Daniel Macrinici, Cristian Cartofeanu, and Shang Gao. 2018. Smart contract applications within blockchain technology: A systematic mapping study. *Telem. Inform.* 35, 8 (2018).
[112] Imran Makhdoom, Mehran Abolhasan, Haider Abbas, and Wei Ni. 2018. Blockchain's adoption in IoT: The challenges, and a way forward. *J. Netw. Comput. Applic.* 125 (2018).
[113] Kamran Mammadzada, Mubashar Iqbal, Fredrik Milani, Luciano García-Bañuelos, and Raimundas Matulevičius. 2020. Blockchain oracles: A framework for blockchain-based applications. In *Proceedings of the International Conference on Business Process Management*. Springer, 19–34.
[114] Christopher Mann and Daniel Loebenberger. 2017. Two-factor authentication for the Bitcoin protocol. *Int. J. Inf. Secur.* 16, 2 (2017), 213–226.
[115] Hartwig Mayer. 2016. ECDSA security in Bitcoin and Ethereum: A research survey. *CoinFaabrik*, PhJune 28 (2016).

[116] Alexander Mense and Markus Flatscher. 2018. Security vulnerabilities in Ethereum smart contracts. In *Proceedings of the 20th International Conference on Information Integration and Web-based Applications & Services*. ACM, 375–380.
[117] Andrew Miller, Ari Juels, Elaine Shi, Bryan Parno, and Jonathan Katz. 2014. Permacoin: Repurposing Bitcoin work for data preservation. In *Proceedings of the IEEE Symposium on Security and Privacy*. IEEE, 475–490.
[118] MITRE. 2018. MITRE ATT&CK. Retrieved from: https://attack.mitre.org/. (2018).
[119] Carlos Molina-Jimenez, Ioannis Sfyrakis, Ellis Solaiman, Irene Ng, Meng Weng Wong, Alexis Chun, and Jon Crowcroft. 2018. Implementation of smart contracts using hybrid architectures with on and off–blockchain components. In *Proceedings of the IEEE 8th International Symposium on Cloud and Service Computing (SC2)*. IEEE, 83–90.
[120] Satoshi Nakamoto et al. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008). https://rb.gy/lk0e98.
[121] Christopher Natoli and Vincent Gramoli. 2016. The blockchain anomaly. In *Proceedings of the IEEE 15th International Symposium on Network Computing and Applications (NCA)*. IEEE, 310–317.
[122] Giang-Truong Nguyen and Kyungbaek Kim. 2018. A survey about consensus algorithms used in blockchain. *J. Inf. Process. Syst.* 14, 1 (2018).
[123] Robert C. Nickerson, Upkar Varshney, and Jan Muntermann. 2013. A method for taxonomy development and its application in information systems. *Eur. J. Inf. Syst.* 22, 3 (2013), 336–359.
[124] Russell O'Connor. 2017. Simplicity: A new language for blockchains. In *Proceedings of the Workshop on Programming Languages and Analysis for Security*. ACM, 107–120.
[125] Orbit. 2017. Orbit-DB: Peer-to-peer Databases for the Decentralized Web. Retrieved from: https://bit.ly/3zwVcYI. (2017).
[126] Pim Otte, Martijn de Vos, and Johan Pouwelse. 2017. TrustChain: A Sybil-resistant scalable blockchain. *Fut. Gen. Comput. Syst.* 107 (2017).
[127] Simone Porru, Andrea Pinna, Michele Marchesi, and Roberto Tonelli. 2017. Blockchain-oriented software engineering: Challenges and new directions. In *Proceedings of the IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*. IEEE, 169–171.
[128] Portswigger. 2021. Latest cryptocurrency security news. Retrieved from: https://bit.ly/38nQL6h. (2021).
[129] ProtocolLabs. 2017. Filecoin: A Decentralized Storage Network. Retrieved from: https://filecoin.io/filecoin.pdf. (2017).
[130] Provable. 2019. Provable Documentation. Retrieved from: https://docs.provable.xyz/. (2019).
[131] Rui Qiao, Sifeng Zhu, Qingxian Wang, and Jie Qin. 2018. Optimization of dynamic data traceability mechanism in Internet of things based on consortium blockchain. *Int. J. Distrib. Sensor Netw.* 14, 12 (2018), 1550147718819072.
[132] Kaihua Qin, Liyi Zhou, Benjamin Livshits, and Arthur Gervais. 2020. Attacking the DeFi ecosystem with flash loans for fun and profit. *arXiv preprint arXiv:2003.03810* (2020).
[133] Emanuel Regnath and Sebastian Steinhorst. 2018. LeapChain: Efficient blockchain verification for embedded IoT. In *Proceedings of the International Conference on Computer-aided Design*. ACM, New York, NY.
[134] Eric Dashofy, Richard N. Taylor, and Nenad Medvidovic. 2009. *Software Architecture: Foundations, Theory, and Practice*. Addison-Wesley Professional.
[135] Sara Rouhani and Ralph Deters. 2019. Security, performance, and applications of smart contracts: A systematic survey. *IEEE Access* 7 (2019), 50759–50779.
[136] Muhammad Saad, Laurent Njilla, Charles Kamhoua, and Aziz Mohaisen. 2019. Countering selfish mining in blockchains. In *Proceedings of the International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 360–364.
[137] Muhammad Saad, Jeffrey Spaulding, Laurent Njilla, Charles Kamhoua, Sachin Shetty, DaeHun Nyang, and David Mohaisen. 2020. Exploring the attack surface of blockchain: A comprehensive survey. *IEEE Commun. Surv. Tutor.* 22, 3 (2020), 1977–2008.
[138] K. Salah, M. H. U. Rehman, N. Nizamuddin, and A. Al-Fuqaha. 2019. Blockchain for AI: Review and open research challenges. *IEEE Access* 7 (2019), 10127–10149. DOI: https://doi.org/10.1109/ACCESS.2018.2890507
[139] Eli Ben Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. 2014. Zerocash: Decentralized anonymous payments from Bitcoin. In *Proceedings of the IEEE Symposium on Security and Privacy*. IEEE, 459–474.
[140] Masashi Sato and Shin'ichiro Matsuo. 2017. Long-term public blockchain: Resilience against compromise of underlying cryptography. In *Proceedings of the 26th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 1–8.
[141] Sarwar Sayeed, Hector Marco-Gisbert, and Tom Caira. 2020. Smart contract: Attacks and protections. *IEEE Access* 8 (2020), 24416–24427.
[142] David Schwartz, Noah Youngs, Arthur Britto, et al. 2014. The Ripple Protocol consensus algorithm. *Ripple Labs Inc White Paper* 5 (2014), 8.
[143] Slowmist. 2021. EOS DApp total loss money by hacked is about. Retrieved from: https://bit.ly/3ynTKX1. (2021).

[144] Yonatan Sompolinsky and Aviv Zohar. 2015. Secure high-rate transaction processing in Bitcoin. In *Proceedings of the International Conference on Financial Cryptography and Data Security*. Springer, 507–527.
[145] C. Stathakopoulous and Christian Cachin. 2017. Threshold signatures for blockchain systems. *Swiss Federal Institute of Technology* (2017). https://rb.gy/rqgvk3.
[146] Harish Sukhwani, José M. Martínez, Xiaolin Chang, Kishor S. Trivedi, and Andy Rindos. 2017. Performance modeling of PBFT consensus process for permissioned blockchain network (hyperledger fabric). In *Proceedings of the IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 253–255.
[147] Don Tapscott and Alex Tapscott. 2017. How blockchain will change addresss. *MIT Sloan Manag. Rev.* 58, 2 (2017), 10.
[148] ThomasKur. 2018. Threats and Countermeasures | Microsoft Docs. Retrieved from: https://bit.ly/38i3Pu6. (2018).
[149] Muoi Tran, Inho Choi, Gi Jun Moon, Anh V. Vu, and Min Suk Kang. 2020. A stealthier partitioning attack against Bitcoin peer-to-peer network. In *Proceedings of the IEEE Symposium on Security and Privacy (IEEE S&P)*.
[150] Muhammad Usman, Ricardo Britto, Jürgen Börstler, and Emilia Mendes. 2017. Taxonomies in software engineering: A systematic mapping study and a revised taxonomy development method. *Inf. Softw. Technol.* 85 (2017), 43–59.
[151] Dejan Vujičić, Dijana Jagodić, and Siniša Ranđić. 2018. Blockchain technology, Bitcoin, and Ethereum: A brief overview. In *Proceedings of the 17th International Symposium INFOTEH-JAHORINA (INFOTEH)*. IEEE, 1–6.
[152] Marko Vukolić. 2017. Rethinking permissioned blockchains. In *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*. ACM, New York, NY, 3–7.
[153] Vyper. 2019. Vyper. Retrieved from: https://bit.ly/38mnVmY. (2019).
[154] Zhiyuan Wan, Xin Xia, and Ahmed E. Hassan. 2021. What do programmers discuss about blockchain? *IEEE Trans. Softw. Eng.* 07 (2021), 1331–1349.
[155] Zhiyuan Wan, Xin Xia, David Lo, Jiachi Chen, Xiapu Luo, and Xiaohu Yang. 2021. Smart contract security: A practitioners' perspective. In *Proceedings of the IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, 1410–1422.
[156] Hai Wang, Yong Wang, Zigang Cao, Zhen Li, and Gang Xiong. 2018. An overview of blockchain security analysis. In *Proceedings of the China Cyber Security Annual Conference*. Springer, 55–72.
[157] Licheng Wang, Xiaoying Shen, Jing Li, Jun Shao, and Yixian Yang. 2019. Cryptographic primitives in blockchains. *J. Netw. Comput. Applic.* 127 (2019), 43–58.
[158] Wenbo Wang, Dinh Thai Hoang, Peizhao Hu, Zehui Xiong, Dusit Niyato, Ping Wang, Yonggang Wen, and Dong In Kim. 2019. A survey on consensus mechanisms and mining strategy management in blockchain networks. *IEEE Access* 7 (2019), 22328–22370.
[159] Jon Whittle, Duminda Wijesekera, and Mark Hartong. 2008. Executable misuse cases for modeling security concerns. In *Proceedings of the 30th International Conference on Software Engineering*. ACM, 121–130.
[160] Claes Wohlin, Per Runeson, Martin Höst, Magnus C. Ohlsson, Björn Regnell, and Anders Wesslén. 2012. *Experimentation in Software Engineering*. Springer Science & Business Media.
[161] Maximilian Wohrer and Uwe Zdun. 2018. Smart contracts: Security patterns in the Ethereum ecosystem and solidity. In *Proceedings of the International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*. IEEE, 2–8.
[162] Jing Wu, Xin Cui, Wei Hu, Keke Gai, Xing Liu, Kai Zhang, and Kai Xu. 2018. A new sustainable interchain design on transport layer for blockchain. In *Proceedings of the International Conference on Smart Blockchain*. Springer, 12–21.
[163] Lijun Wu, Kun Meng, Shuo Xu, Shuqin Li, Meng Ding, and Yanfeng Suo. 2017. Democratic centralism: A hybrid blockchain architecture and its applications in energy internet. In *Proceedings of the IEEE International Conference on Energy Internet (ICEI)*. IEEE, 176–181.
[164] Karl Wüst and Arthur Gervais. 2018. Do you need a blockchain? In *Proceedings of the Crypto Valley Conference on Blockchain Technology*. IEEE, 45–54.
[165] Quanqing Xu, Zhiwen Song, Rick Siow Mong Goh, and Yongjun Li. 2018. Building an Ethereum and IPFS-Based decentralized social network system. In *Proceedings of the IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 1–6.
[166] Xiwei Xu, Qinghua Lu, Yue Liu, Liming Zhu, Haonan Yao, and Athanasios V. Vasilakos. 2019. Designing blockchain-based applications a case study for imported product traceability. *Fut. Gen. Comput. Syst.* 92 (2019), 399–406.
[167] Xiwei Xu, Cesare Pautasso, Liming Zhu, Vincent Gramoli, Alexander Ponomarev, An Binh Tran, and Shiping Chen. 2016. The blockchain as a software connector. In *Proceedings of the 13th Working IEEE/IFIP Conference on Software Architecture (WICSA)*. IEEE, 182–191.
[168] Xiwei Xu, Cesare Pautasso, Liming Zhu, Qinghua Lu, and Ingo Weber. 2018. A pattern collection for blockchain-based applications. In *Proceedings of the 23rd European Conference on Pattern Languages of Programs*. ACM.
[169] Xiwei Xu, Ingo Weber, Mark Staples, Liming Zhu, Jan Bosch, Len Bass, Cesare Pautasso, and Paul Rimba. 2017. A taxonomy of blockchain-based systems for architecture design. In *Proceedings of the IEEE International Conference on Software Architecture (ICSA)*. IEEE, 243–252.

[170] Dylan Yaga, Peter Mell, Nik Roby, and Karen Scarfone. 2019. Blockchain technology overview. *arXiv preprint arXiv:1906.11078* (2019).
[171] Kazuhiro Yamashita, Yoshihide Nomura, Ence Zhou, Bingfeng Pi, and Sun Jun. 2019. Potential risks of hyperledger fabric smart contracts. In *Proceedings of the IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*. IEEE, 1–10.
[172] Jinhong Yang, Md Mehedi Hassan Onik, Nam-Yong Lee, Mohiuddin Ahmed, and Chul-Soo Kim. 2019. Proof-of-familiarity: A privacy-preserved blockchain scheme for collaborative medical decision-making. *Appl. Sci.* 9, 7 (2019), 1370.
[173] Zhe Yang, Kan Yang, Lei Lei, Kan Zheng, and Victor C. M. Leung. 2018. Blockchain-based decentralized trust management in vehicular networks. *IEEE Internet Things J.* 6, 2 (2018), 1495–1505.
[174] Wei Yin, Qiaoyan Wen, Wenmin Li, Hua Zhang, and Zhengping Jin. 2018. An anti-quantum transaction authentication approach in blockchain. *IEEE Access* 6 (2018), 5393–5401.
[175] Mahdi Zamani, Mahnush Movahedi, and Mariana Raykova. 2018. RapidChain: Scaling blockchain via full sharding. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. 931–948.
[176] Rui Zhang, Rui Xue, and Ling Liu. 2019. Security and privacy on blockchain. *arXiv preprint arXiv:1903.07602* (2019).
[177] Qiuhong Zheng, Yi Li, Ping Chen, and Xinghua Dong. 2018. An innovative IPFS-based storage model for blockchain. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. IEEE, 704–708.
[178] Zibin Zheng, Shaoan Xie, Hongning Dai, Xiangping Chen, and Huaimin Wang. 2017. An overview of blockchain technology: Architecture, consensus, and future trends. In *Proceedings of the 6th International Congress on Big Data*. IEEE, 557–564.
[179] Zibin Zheng, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang. 2018. Blockchain challenges and opportunities: A survey. *Int. J. Web Grid Serv.* 14, 4 (2018), 352–375.
[180] Zibin Zheng, Shaoan Xie, Hong-Ning Dai, and Huaimin Wang. 2016. Blockchain challenges and opportunities: A survey. *Work Pap.–2016* (2016). https://rb.gy/btg0vl.
[181] Lin Zhong, Qianhong Wu, Jian Xie, Jin Li, and Bo Qin. 2019. A secure versatile light payment system based on blockchain. *Fut. Gen. Comput. Syst.* 93 (2019), 327–337.
[182] Liehuang Zhu, Yulu Wu, Keke Gai, and Kim-Kwang Raymond Choo. 2019. Controllable and trustworthy blockchain-based cloud data management. *Fut. Gen. Comput. Syst.* 91 (2019), 527–535.
[183] Weiqin Zou, David Lo, Pavneet Singh Kochhar, Xuan-Bach D Le, Xin Xia, Yang Feng, Zhenyu Chen, and Baowen Xu. 2019. Smart contract development: Challenges and opportunities. *IEEE Trans. Softw. Eng.* 47, 10 (2019).

Received April 2021; revised September 2021; accepted November 2021