Received April 7, 2021, accepted May 14, 2021, date of publication May 19, 2021, date of current version May 28, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3081998

Exploring Sybil and Double-Spending Risks in Blockchain Systems

MUBASHAR IQBAL AND RAIMUNDAS MATULEVIČIUS
Institute of Computer Science, University of Tartu, 51009 Tartu, Estonia

Corresponding author: Mubashar Iqbal (mubashar.iqbal@ut.ee)

This work was partially supported by the ERASMUS+ sectoral alliance program on *A Blueprint for sectoral cooperation on blockchain skill and development* (CHAISE), under grant no. 621646-EPP-1-2020-1-FR-EPPKA2-SSA-B.

ABSTRACT The first step to realise the true potential of blockchain systems is to explain the associated security risks and vulnerabilities. These risks and vulnerabilities, exploited by the threat agent to affect the valuable assets and services. In this work, we use a security risk management (SRM) domain model and develop a framework to explore two security risks – *Sybil* and *Double-spending* – that are observed and considered most concerning security risks within blockchain systems. The framework illustrates the protected assets or assets to secure, the classification of threats that the attacker can trigger using Sybil attack, the identification of threats that cause Double-spending, the vulnerabilities of identified threats, and their countermeasures. We evaluated a newly built framework by exploring Sybil and Double-spending risks in Ethereum-based healthcare applications. We also recognise the various other security and implementation challenges of blockchain that hinder the acceptance of blockchain-enabled solutions. Furthermore, we discuss the permissioned blockchain systems making an appearance in industry-level enterprises and how permissioned blockchain systems control these challenges. We conclude the paper and outline the future work that aims to build an ontology-based blockchain security reference model. The results of this work could help blockchain developers, practitioners, and other associated stakeholders to communicate about Sybil and Double-spending risks, what security countermeasures should be introduced, and what security and implementation challenges are emerging in blockchain systems.

INDEX TERMS Blockchain, blockchain systems, sybil attack, double-spending, security risk management, blockchain emerging challenges.

I. INTRODUCTION

Blockchain is a decentralised, distributed, and immutable ledger technology [1]. Blockchain technology operates over a peer-to-peer (P2P) network and distributes a ledger every time on an entire P2P network when a new block (or transaction) occurs [13], [15]. A ledger contains a certain and verifiable record of every single transaction ever made [2]. The advent of Bitcoin magnifies the research in the blockchain domain. The turing complete language-based smart contracts in blockchain made a prominent contribution to the development of blockchain technology. Smart contracts' ultimate goals are to eliminate trusted intermediaries, less human intervention, reduce enforcement costs, prevent intentional or unintentional fraud and security risks. Blockchain technology is continuously penetrating various fields, and the involvement of monetary assets raises security concerns, mainly when the attacker may steal the assets or damage the system. For example, in a decentralised autonomous organisation (DAO) attack, the attacker exploited the reentrancy vulnerability in Ethereum smart contract and gained control on $60 million Ethers [3], [4].

Blockchain technology promises to overcome security challenges, enhance data integrity, and transform the transacting process into a decentralised, transparent, and immutable manner. Thus, security plays an important role to guarantee blockchain acceptability. Blockchain systems are considered to be less vulnerable because of decentralised consensus, immutable ledger, and cryptography. For example, the study [5] illustrated how the data tampering risk could be mitigated by using blockchain, and Xiaoding *et al.* [121] implemented the blockchain-based authentication mechanism to

The associate editor coordinating the review of this manuscript and approving it for publication was Alessandra De Benedictis.

overcome the limitations of single-factor authentication in the industrial internet of things (IIoT). However, there exist security risks (e.g., *Sybil, Double-spending* and others) that appear within the blockchain systems [6].

A. MOTIVATION

The studies [122], [123] investigated the security risks of traditional applications, and developed defence strategies to defend applications from various security risks. In a similar context, to realise the true potential of blockchain systems, the first step is to explore and clarify the security risks that could appear. In previous work [6], we identified the *Sybil* and *Double-spending* are the most concerning security risks within blockchain systems. The attacker could exploit these security risks, affect the valuable assets and services of blockchain systems. Therefore, we decided to focus on Sybil and Double-spending risks to examine them further in detail.

Sybil attack is a network attack and well-known in the context of P2P networks [7], where an attacker can forge or create numerous fake identities to gain a considerable influence on the network. The related studies (Section II-D) discuss the Sybil attack in blockchain systems on a generic level (e.g., the attacker creates Sybil identities and damages the reputation system). On the contrary, the threats spectrum of Sybil attack is much vast, and the attacker has various motives to achieve by using the Sybil attack (Table 7).

Double-spending is a data consistency attack, and it happens when spending the same digital money (or digital asset) twice. In general, Double-spending is a technique that can be used to deceive someone about the state of a transaction. For instance, on a date (5th of Jan 2019), a total of 219,500 ETC (worth $1.1 million at the time of the attack) tokens were double-spent as reported by a Coinbase cryptocurrency exchange [8]. The 51% attack is mostly discussed as a cause of Double-spending [23], [34]. In contrast, several other approaches can trigger Double-spending (Table 8).

Nowadays, various organisations build their customised blockchain systems to fulfil their specific needs [91], [92]. Such systems are prone to different security risks, including Sybil and Double-spending risks. In a similar context, the developers are not aware of certain security risks when developing blockchain-based applications [32], [33]. The key explanation for this is that there is no corresponding mechanism (or framework) to explore or prevent related security risks. Furthermore, a few studies [35], [36] provide a tool to select a blockchain platform for building blockchain-based applications but did not discuss the associated security risks that could emerge in them.

B. CONTRIBUTIONS

We follow the security risk management (SRM) domain model [9], [10] and develop a framework (Table 7 & 8) to explore Sybil and Double-spending risks in blockchain systems. The framework helps to communicate about Sybil and Double-spending risks to blockchain developers, practitioners, and other associated stakeholders. Among other SRM approaches [11], [12], the SRM domain model helps to explore *assets-related, risk-related*, and *risk treatment-related* concepts. The main contributions of this research are as follows:

- Identifying threats that the attacker can trigger by using Sybil attack
- Identifying threats that can enable the attacker to trigger Double-spending
- Framework based on the SRM domain model to explore Sybil and Double-spending risks of blockchain systems
- Applying framework to explore Sybil and Double-spending risks in Ethereum-based healthcare applications
- Emerging challenges of blockchain and permissioned blockchain systems

We use the term blockchain systems that is combining the blockchain platforms (e.g., Bitcoin, Ethereum, Hyperledger Fabric) and blockchain-based applications (e.g., decentralised applications (dApps)). The rest of the paper is structured as follows: Section 2 provides background and discusses the blockchain, research method, overview of Sybil and Double-spending risks, and related work. Section 3 presents the analysis of Sybil and Double-spending risks and illustrates the framework. Section 4 gives an overview of framework use. Section 5 addresses the emerging challenges of blockchain, and Section 6 provides an overview of permissioned blockchain systems. In Section 7, we confer future work and threats to validity. Section 8 concludes the paper.

II. BACKGROUND

This section presents the overview of blockchain technology, discusses the research method, briefly talks about the Sybil and Double-spending risks, and provides the related work.

A. BLOCKCHAIN

Blockchain eliminates trusted intermediaries in a transactional process and records transactions in a decentralised distributed ledger. A blockchain ledger is a chain of blocks (Fig. 1) where each block connects to a previous block by a unique cryptographic hash. The first block in a blockchain is a genesis block, and every block has a header and body.

The format of a block is discussed in Table 1. Block header includes a unique block hash, a previous block hash, Merkle root, block version, nonce, timestamp, and difficulty target. Block body contains a valid list of transactions that are hashed and ordered as a Merkle tree.

Table 2 provides a comparison of blockchain platforms. Blockchain platforms can be classified as permissionless (e.g., Bitcoin, Ethereum) or permissioned (e.g., Hyperledger Fabric (HLF), Corda) [35]. In permissionless blockchains, anybody around the world can join the network. The network participants do not require permissions to participate in the consensus or executing a transaction. Also, the transactions are publicly visible to everyone. In contrast, in permissioned blockchains, only pre-verified nodes can join the network,

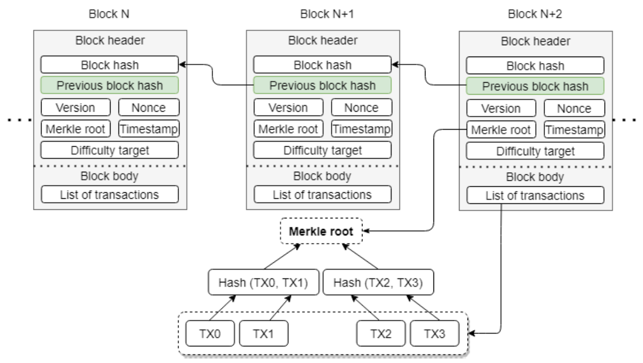

FIGURE 1. Blockchain and block structure.

TABLE 1. Blockchain block format.

Block element	Detail
Block hash	A unique cryptographic hash of a block.
Previous block hash	A unique cryptographic hash of a previous block.
Version	Indicates the version of the block.
Nonce	A unique random counter.
Merkle root	The hash of Merkle tree.
Timestamp	The creation time of the block.
Difficulty target	Indicates a mining difficulty in PoW-based consensus
Transactions	A list of transactions stored in Merkle tree format.

the access control layer controls the network participants operations, and transaction visibility is restricted [13].

The advent of smart contracts (SC) introduces value exchange in P2P and decentralised manners. A smart contract is an autonomous computer program [14] written in a turing-complete programming language [108] (e.g., Solidity, GO, Java, C++). The SC constitutes a digital contract in blockchain to store data and execute automatically [15] when certain conditions meet. For example, the Ethereum platform provides *Solidity* programming language to write SC and to build decentralised applications (dApps) [14]. In HLF, SC is known as Chaincode and performs actions according to the coded terms in a contract. Similarly, other blockchain platforms (e.g., Corda, EoS, Stellar) have SC to reach contractual agreements in a digital realm [4].

Blockchain uses a decentralised consensus mechanism to maintain the state and immutability of the ledger. Bitcoin and Ethereum use Proof of Work (PoW) consensus. PoW is a computational rich energy-waste consensus strategy where special nodes, called miners, define the state of the ledger by solving the complex cryptographic puzzle. In contrast, Proof of Stake (PoS) is an energy-efficient consensus strategy [16] where miners become validators [14] and lock a certain amount of cryptocurrency to participate in the consensus. The blockchain platforms NXT and PeerCoin are using PoS consensus, and Ethereum is moving to PoS consensus. HLF uses practical byzantine fault tolerance (PBFT) consensus that tolerates byzantine faults (e.g., malicious nodes) [85]–[87]. PBFT is an energy-efficient and effective consensus mechanism for high-throughput transactions. There exist other consensus mechanisms, for example, Delegated Proof of Stake (DPoS), Proof of Authority (PoA), Proof of Reputation (PoR), Proof of Spacetime (PoSt), and new are appearing. These consensus mechanisms are fault-tolerant and achieve mutual agreement on a single state of the distributed ledger [23].

B. RESEARCH METHOD

In our previous study [6], we conduct a systematic literature review by following the guidelines of [93] to identify security risks in blockchain systems. First, we identify the security risks of centralised applications that are mitigated by using blockchain-enabled solutions (e.g., data tampering risk). Second, we identify the security risks that appear within blockchain systems (e.g., Sybil and Double-spending risks). The results [6] show that *Sybil* and *Double-spending* risks are the utmost concerns in blockchain systems. Therefore, we consider these two security risks to explore further.

In this work, we follow the SRM domain model (Fig. 2) [9], [10] to develop a framework to explore Sybil and Double-spending risks within blockchain systems. The SRM domain model enables a systematic approach to analyse both security risks in the perspective of assets-related, risk-related, and risk treatment-related concepts.

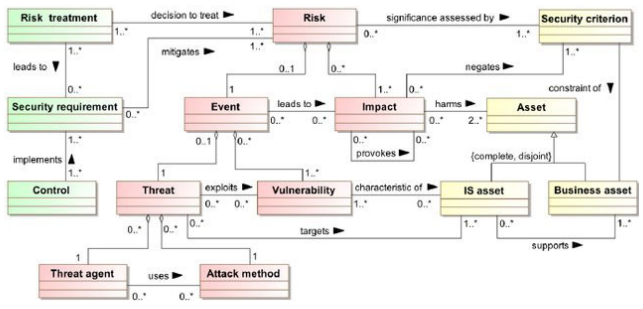

FIGURE 2. The security risk management domain model [9], [10].

The asset can be classified as a system/information system asset or a business asset. The business asset has value, and the system asset supports the business asset. Security criteria (C - Confidentiality, I - Integrity, and A - Availability) are business asset constraints and distinguish the security needs. In risk-related concepts, the risk is a combination of risk event and impact. Impact harms the asset and negates the security criteria. The risk event constitutes the threat and one or more vulnerabilities. The threat targets the system asset, and it is triggered by the threat agent. The threat agent[1] uses an attack method and exploits the vulnerability. The risk treatment-related concepts present decisions to treat security risks by defining security requirements. Security requirements are implemented in the security controls (e.g., countermeasures implement the requirements).

In [17], the authors perform a survey and identify six different countermeasure strategies (Table 3). These strategies utilise in different blockchain systems to secure them against Double-spending and selfish mining risks. These strategies are based on the blockchain design parameters and

[1]In this paper, a general term "attacker" is used instead of the more specific "threat agent".

TABLE 2. Comparison of blockchain platforms.

	Bitcoin	Ethereum	HLF	Corda	EoS	Ripple	Stellar
Type	Permissionless	Permissionless	Permissioned	Permissioned	Permissioned	Permissioned	Both
Consensus	PoW	PoW, *PoS*	PBFT, CFT	Validity, Uniqueness	DPoS	Probabilistic voting	Stellar protocol
Participation	Open	Open	Member-only	Member-only	Member-only	Member-only	Open, Member-only
Read/Write Operation	Any participant	Any participant	Verified member	Verified member	Verified member	Verified member	Any participant, Verified member
Smart contract	Yes	Yes	Yes	Yes	Yes	No	Yes
SC Language	Script	Solidity	GO, Java	Kotlin	C++	- - -	Node JS
Cryptocurrency	Bitcoin (BTC)	Ether (ETH)	- - -	- - -	EOS	XRP	XLM
Throughput	7 TPS	15 TPS	3500+ TPS	1678 TPS	4000+ TPS	1500+ TPS	1000+ TPS
Scalable	Difficult	Difficult	Moderate	Moderate	Moderate	Moderate	Moderate
Nodes capacity	Unlimited	Unlimited	Moderate	Limited	Unlimited	Limited	Moderate
Confidentiality	No	No	Yes	Yes	Yes	Yes	Yes
Transparency	High	High	Moderate	Low	Moderate	Low	Moderate
Open source	Yes	Yes	Yes	Yes	Yes	No	Yes
Governance	- - -	Ethereum Developers	Linux Foundation	R3 Consortium	EOSIO Core Arbitration Forum (ECAF)	Ripple Labs	Stellar Development Foundation
Industry focus	Cryptocurrency	Enterprise	Enterprise	Financial	Enterprise	Financial	Financial
Adoption	High	High	High	Moderate	Moderate	Moderate	Low

TABLE 3. Countermeasure strategies that utilise in blockchain systems to secure them against Double-spending and selfish mining risks [17].

Strategy	Detail
Monitoring	Countermeasures that establish to monitor user activity based on time, usage, tasks, and operations.
Alert forwarding	Implementation of communication-based alert system that aims to notify attacks to the network nodes.
Alert broadcasting	Provides public system information to recipient addresses, for example, private details relating to the attackers.
Inform	Techniques to analyse blockchain security based on raw facts, models, figures, processes, and mathematical algorithms.
Detection	Inspection that detect and notify about the abnormalities and misbehaviour in the blockchain system.
Conceptual research design	Belongs to conceptual research proposal, predictive models, and theoretically proposed techniques.

implementation of security solutions. We use these strategies to classify the countermeasures of Sybil and Double-spending risks.

Algorithm 1 Sybil Attack

```
$H \leftarrow$ Honest nodes;
$S \leftarrow$ Sybil nodes;
$A \leftarrow$ Attacker node;
while ($true$) do
    $A$ Creates $S$;
    $A$ Connects $S$ with $H$;
    $A$ Gains fraction of the system $\leftarrow \Delta$;
    if ($\Delta == true$) then
        $A$ Uses attack method;
        $A$ Triggers threat;
        $A$ Damages reputation system;
    end
end
```

C. SECURITY RISKS

The demand for technology is growing because it provides ease, but the security risks that cyberspace faces [17], [18], [31], [124], [125] cannot be overlooked. This section presents a brief overview of Sybil and Double-spending risks that are most concerning security risks [6] within blockchain systems.

1) SYBIL ATTACK

In [18], Douceur discussed the **Sybil attack** on P2P systems. The P2P systems do not rely on a central trusted party chain of trust to verify the identity of each participant node, also relatively cheap to generate identities on P2P systems that treat equally on the network [26], [94].

Algorithm 1 explains the procedure of a Sybil attack that undermines a P2P network [18]. The Sybil attack combines the honest (H), Sybil (S), and attacker (A) nodes. To initiate the attack, the attacker creates numerous Sybil nodes and connects with the honest nodes that disconnect genuine connections of honest nodes with other honest nodes on the P2P network. The attacker takes control over the P2P network when he gains a disproportionately large influence on the network (Δ). Eventually, the attacker uses Sybil nodes and attack method to trigger various threats that damage the reputation system of a P2P network.

In traditional P2P systems, the Sybil attack imposes various threats and subverts the reputation of a P2P network. For example, the attacker uses Sybil identities to pollute the BitTorrent distributed hash table (DHT) routing to trigger delays, connection slot consumption, invalid file contents downloads, and bandwidth exhaustion [19]. Similarly, the attacker can use the clone node attack to target static wireless sensor networks (WSNs) [124] by taking control of *node n* and replicating it throughout the network. On a successful clone node attack, the attacker can initiate a Sybil attack, selective forwarding attacks, incorrect data injection, protocol interruptions and traffic jams [124].

Sybil attacks are hard to prevent [7], however, there exist preventive measures to increase protection against Sybil attack. The authors [19] present a reputation-based scheme 'GOLF' to identify Sybil nodes on BitTorrent DHT based on patterns in IP addresses. Numan *et al.* [124] perform a literature review and assemble the cloned node detection schemes along with their drawbacks and challenges. Furthermore, the study [7] categorised three different mechanisms to defend P2P systems against Sybil attacks: (1) Trusted certification (e.g., centralised or distributed certification using cryptographic primitives), (2) resources testing (e.g., IP testing, network coordinates, requiring clients to solve puzzles), and (3) social network techniques (e.g., SybilGuard, SybilLimit, SybilInfer, vote aggregation, and GateKeeper).

Blockchain systems run over the P2P network; therefore, it is possible to run multiple fake nodes [27]. Additionally, blockchain systems include valuable digital assets that motivate the attackers to execute this attack. Once fake identities gain recognition in the blockchain system, the attacker interrupts the flow of information, out-votes (or block) the honest nodes, and refuses to receive or transmit information [24], [94]. The threats spectrum is increasing on blockchain systems, and the attacker has different motives to carry out this attack. In section III-A, Sybil attack-based threats are addressed in detail.

2) DOUBLE-SPENDING

Double-spending is a risk of digital currency where the attacker can spend the same currency twice to gain monetary benefits [31]. For instance, the attacker changes the transaction state and spends the same transaction twice [48]. The risk of Double-spending negates the integrity of the ledger. Several threats exist that can cause Double-spending, for example, Sybil-based Double-spending, 51% attack, etc. In section III-B, these threats are addressed in detail.

Algorithm 2 illustrates the procedure of Double-spending in the case of 51% attack. The attack constitutes the honest nodes *(H)*, honest chain *(HC)*, attacker node *(A)*, attacker chain *(AC)*, and the attacker computing-power *(ACP)*. To initiate the attacker, the attacker forks a private chain from an honest chain. Next, he gains 51% or more computing-power in blockchain and creates two conflicting transactions $(TX1, TX2)$. The attacker sends one transaction in an honest chain $(TX1 \rightarrow HC)$ paying a merchant for some goods and another double-spend transaction to himself in his chain $(TX2 \rightarrow AC)$. The attacker starts mining blocks on his chain and generates blocks quicker than the honest nodes by using his computing-power. Once the attacker chain length is greater than the honest chain length $(AC > HC)$, the attacker chain becomes valid and honest nodes adopt it based on the longest chain rule. Thus, it makes the double-spend transactions valid, and the attacker receives his spent funds back to himself.

Algorithm 2 Double-Spending by 51% Attack

```
$H \leftarrow$ Honest nodes;
$HC \leftarrow$ Honest chain;
$A \leftarrow$ Attacker node;
$AC \leftarrow$ Attacker chain;
$ACP \leftarrow$ Attacker computing-power;
while (true) do
    $A \rightarrow$ Forks private chain $AC$ from $HC$;
    if ($ACP \geq 51\%$) then
        if ($A \rightarrow$ Creates conflicting transactions) then
            $TX1 \rightarrow HC$;
            $TX2 \rightarrow AC$;
            $A \rightarrow$ Starts mining $AC$ ;
            if ($AC.length > HC.length$) then
                $AC$ becomes Valid;
                $H$ adopt $AC$;
                $A$ gets spent funds back;
            end
        end
    end
end
```

D. RELATED WORK

The acceptance of blockchain technology continues to increase in various fields such as healthcare, resource monitoring, digital rights management, financial services, smart vehicles, supply chain, IoT, etc. [6]. For example, the authors [126] use the blockchain in an IoT network to secure battery life data gathered from IoT sensors, and [136] blockchain-assisted secure data sharing model for IoT-based smart industries. The study [32] builds the patient medical health record system using blockchain, and [33] provides a blockchain-enabled tamper-proof insurance claim system. The growth of blockchain-based solutions maximises the research of blockchain security. There exist a few studies that evaluated the security of various blockchain systems. The related works fall short in various directions that the current work addresses (Table 4).

Nicolas *et al.* [17] perform a systematic literature study to explore defensive strategies of blockchain systems that protect against Double-spending and selfish mining. The study identifies the six defensive (countermeasure) strategies, for instance, monitoring, alert forwarding, alert broadcasting, inform, detection, and conceptual research design (Table 3). The authors focus on the countermeasures of Double-spending and selfish mining that are classified in these six countermeasure strategies. In contrast, our study provides a framework based on the SRM domain model to explore Sybil and Double-spending risks in blockchain systems. Moreover, we focus on the threats of Sybil and Double-spending risks, the assets to secure in blockchain systems, the vulnerabilities, and what are the countermeasures for risk-treatments.

In [20], the authors identify different attacks in blockchain systems, mainly emphasising attack surfaces. For example, the security attacks on blockchain cryptographic constructs, distributed architecture (P2P network), and blockchain-based applications. The study does not discuss any attack in detail or their associated vulnerabilities. Another work [21] that classifies security threats in blockchain technology. Similarly, this study does not explore the threats in detail, their vulnerabilities or affected assets. Conversely, we present a framework to explore Sybil and Double-spending risks detailing threats, vulnerabilities, affected assets, countermeasures, and countermeasures strategies.

Pinzón *et al.* [22] present the Double-spending attack model with a time advantage on the Bitcoin blockchain.

TABLE 4. Mapping of related work that explores the security risks of blockchain systems. (x) denotes that the stated characteristic was addressed, while (–) denotes that it was not. For example, none of the related work discussed what assets to secure, the threats associated with the risks, and the SRM domain model.

	Asset	Risk	Threat	Vulnerability	Countermeasure	Defense strategy	SRM domain model	Framework
Nicolas et al. [17]	–	x	–	–	x	x	–	x
Saad et al. [20]	–	x	–	–	x	–	–	–
Mosakheil et al. [21]	–	x	–	x	–	–	–	–
Pinzon et al. [22]	–	x	–	–	–	–	–	–
Sayeed et al. [23]	–	x	–	–	–	–	–	–
Zhang et al. [24]	–	x	–	–	x	–	–	–
Current work	x	x	x	x	x	x	x	x

The paper introduces the two different time advantage-based approaches and algorithmic comparison to show the probability of both approaches to carry Double-spending. The research work does not present attack scenarios, vulnerabilities, affected assets, or viable risk treatments. In [23], authors assess blockchain consensus mechanisms against the 51% attack, which could cause Double-spending. The study managed a comparison of different consensus mechanisms along with their weaknesses. Zhang *et al.* [24] present the attack model that combines a Double-spending attack with a Sybil attack in the Bitcoin network. The study exemplifies the attack states, the success probability of the proposed combined attack, and suggestions to improve the Bitcoin network.

The related works mentioned earlier rely primarily on identifying the security risks associated with the blockchain systems (Table 4). None has explained the security risks by addressing their associated threats, vulnerabilities, assets affected or what assets to protect, and countermeasures. These related studies fail to report on *security risk management* of blockchain systems because they do not follow any SRM domain model. Furthermore, only Nicolas *et al.* [17] built the framework for systematic evaluation of countermeasures against security risks. In this work, we utilise the SRM domain model to develop a framework and conduct security risk management of Sybil and Double-spending risks, report the threats and vulnerabilities, the assets to be secured or classify affected assets within blockchain systems and countermeasures for risks treatments.

III. ANALYSIS OF SYBIL AND DOUBLE-SPENDING RISKS

In this section, we develop a framework (Table 7 & 8) using the SRM domain model to explore Sybil and Double-spending risks. The framework constitutes the blockchain characteristic, risk-related, assets-related, and risk-treatment related concepts.

A. SYBIL ATTACK

The attacker can trigger different threats to gain benefits in a blockchain system by a Sybil attack. The threats available in Table 7 are addressed in detail here.

1) BREAK CONSENSUS PROTOCOL ATTACK

Shards-based blockchain systems (e.g., Elastico [39]) process transactions in parallel to overcome transaction throughput and network scalability limitations of PoW. These systems are vulnerable to Sybil-based break consensus protocol (BCP) attack [25] when an attacker uses the Sybil nodes to disrupt the shards-based consensus process. Shard-based consensus protocol uses PoW to create verifiable node IDs to participate in the consensus process. A valid node in the network could act as an attacker and generate Sybil IDs in target shard(s). The Sybil IDs enable the attacker to break the intra-shard consensus protocol and prevent the insertion of transactions in a blockchain.

TABLE 5. Numerical analysis of BCP attack [25].

Computing-power	# of shards	# of nodes	P_{BCP}
$\leq$ [25%]	at most 16	at least 600	$\leq 10^{-4}$
[33% − 53%]	at most 16	at least 600	≥ 0.8
$\geq$ [56%]	at most 16	at least 600	1

TABLE 6. Numerical analysis of GFT attack [25].

Computing-power	# of shards	# of nodes	P_{GFT}
$\leq$ [25%]	at most 16	at least 600	0
[33% − 53%]	at most 16	at least 600	≤ 0.005
[56%]	at most 16	at least 600	≤ 0.001
$\geq$ [66%]	at most 16	at least 600	≥ 0.75

When using PoW to generate verifiable nodes IDs, the existing shard-based protocol assumes that the network nodes have a uniform computing-power. In contrast, the computing-power of network nodes does not hold uniformity in PoW. Consequently, the attacker exploits this vulnerability in a shard-based consensus protocol. The successful BCP attack impacts the IDs generation process of nodes, preventing the insertion of transactions in a blockchain, and harms the business assets. For example, the BCP attack negates the integrity of IDs generation and consensus process, and transaction availability.

The authors [25] perform numerical analysis to identify the success probability of BCP attack (P_{BCP}) by using different settings of computing-power, the number of shards, and network nodes (Table 5). For example, if an attacker gains 25% computing-power, the system has at most 16 number of shards, and at least 600 number of nodes, then P_{BCP} is $\leq 10^{-4}$. Proportionally, more computing-power increases the P_{BCP}, in this manner, when computing-power is between 33%-53% the P_{BCP} is ≥ 0.8 and P_{BCP} becomes 1 when computing-power in $\geq 56\%$.

The simulation to prevent BCP attack depends on different settings of shards, nodes and their computing-power in the blockchain system [25]. For example, monitor computing-power of nodes and if some node starts gaining computing-power in the blockchain system, then either restrict his computing-power or proportionally increase the number of shards and nodes to prevent BCP attack.

TABLE 7. Framework based on SRM domain model to explore Sybil attack.

Blockchain characteristic		Risk-related concept			Asset-related concept		Risk treatment concept	
Type	**Consensus**	**Threat**	**Vulnerability**	**Impact**	**System asset**	**Business asset**	**Countermeasure**	**Strategy**
Shards-based permissionless	PoW, PBFT, Intra-committee consensus	Break consensus protocol [25]	Computing-power of nodes do not hold uniformity in PoW	Damage node IDs generation process, prevent insertion of transactions, and losing the integrity of consensus process	Nodes, P2P Network, Intra-committee consensus protocol, Computing power	Consensus process (I), ID generation (I), Transaction (A)	Monitor computing-power [25]	Monitoring
							Number of shards [25]	Conceptual
							Increase network nodes [25]	Conceptual
Shards-based permissionless	PoW, PBFT, Intra-committee consensus	Generate fake transaction [25]	Assumption of only verifiable nodes participate in consensus	Losing the integrity of transaction and ledger	Nodes, P2P Network, Consensus process	Transaction (I), Ledger (I)	Monitor computing-power [25]	Monitoring
							Number of shards [25]	Conceptual
							Increase network nodes [25]	Conceptual
Customised permissionless	Trustworthiness of nodes	Tampering nodes reputation [26]	Calculation of nodes reputation score combining Sybil nodes	Damage or lower the reputation of honest nodes	Nodes, P2P Network, Subjective work graph	Nodes reputation (I)	Accounting-based algorithm (NetFlow) [26]	Detection
Permissionless	PoW	Nodes isolation attack [21], [23], [27], [47], [94], [97], [98]	Lack of computing power	Weaken the P2P network	Nodes, P2P Network, Computing power	Network reputation (I)	Increase computing power [23]	Conceptual
							Monitor computing-power [23]	Monitoring
			BGP does not validate routing origin	Obstruct block propagation to affect transaction verification	Nodes, P2P Network, Consensus	Transaction verification (I)	Monitor round-trip time [47], [98]	Monitoring
			No proper authentication of nodes	Losing the integrity and availability of transaction verification	Nodes, P2P Network, Network reputation, Transaction	Transaction verification (I)	Network joining fee [94]	Detection
							Validating node connection [94]	Detection
							Monitoring nodes behavior [94]	Monitoring
Customised permissionless	Respect Score	Routing table insertion [27], [99]	Sybil nodes can update the routing table information	Breaks routing system, modify route message, block access to the transaction, effect voting outcome, and interrupt network services	Nodes, P2P network, Nodes Identifier, Routing table	Message route (I, A), Routing table (I)	Registration of nodes [27]	Forwarding
							Monitor activities of node [27]	Monitoring
							Reward of respect score [27]	Detection
							Route prediction model [99]	Broadcasting
Permissionless	PoW, PoS, DPoS	Sybil-based linking [28]	Pairing with the same users multiple times during mixing	Negates the confidentiality of user identify and transaction	Mixer, Mixing process, Users, Transactions	User Identity (C), Transaction (C)	Non-refundable deposit to create node identity [28]	Detection
							Time locking on funds usage [28]	Inform
							Coin-age [28]	Broadcasting
Permissionless	PoW, PoS, DPoS	Sybil-based DoS [20], [28], [100], [101], [104]	Sybil nodes refuse pairing with honest nodes	Disrupt the functioning of mixer	Mixer, Mixing protocol, Users,	Mixer functioning (A)	Non-refundable deposit to create node identity [28]	Detection
			Sybil nodes can participate in mining	Halt mining process	Nodes, P2P network, Mining protocol	Mining process (A), Mining pool (A)	Use computational constraint-based techniques [28]	Conceptual
			Dusting transactions	Congest blockchain network	Transactions, P2P network	Services (A)	Anti-dust model [104]	Detection

2) GENERATE FAKE TRANSACTION ATTACK

Similar to the BCP attack, the Sybil-based generate fake transaction (GFT) attack is possible on shards-based blockchain systems [25]. This attack aims to use Sybil nodes and create fake (or invalid) transactions in blocks to invalidate the state of the ledger or simply corrupting the transactions. Unlike the BCP attack, in the GFT attack, the attacker uses Sybil nodes to control and manipulate the shard-based consensus process to reach a final consensus on fake transactions to add them into the block. The assumption of only verifiable nodes participating in the consensus process allows the attacker to exploit the shard-based consensus process. The successful GFT attack harms the integrity of transaction and ledger.

To summarise, the numerical analysis in [25] identifies the success probability of the GFT attack (P_{GFT}). The results show that (Table 6) the attacker requires relatively higher computing-power to initiate a GFT attack. Similar to the BCP attack, to prevent GFT attack, monitor computing-power of nodes and use different settings of shards and nodes depending on the computing-power of nodes.

3) TAMPERING NODES REPUTATION

TrustChain [26] is a blockchain-based tamper-proof and scalable data structure specifically designed to create reputation-based distributed trust. TrustChain includes one transaction per block and together form a directed acyclic graph (Fig. 3) where each transaction block has two incoming and two outgoing pointers. The system detects the violation of this rule and considers it fraud that lowers the reputation of the node. Meanwhile, other nodes involve and reach a consensus on the transaction. Furthermore, the system uses a subjective

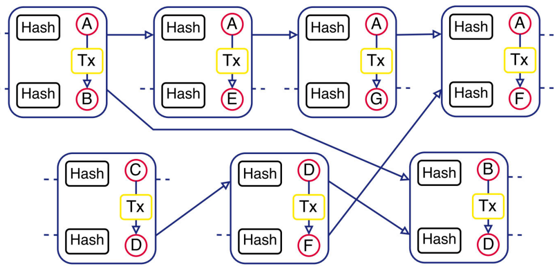

FIGURE 3. TrustChain data structure to record transactions [26].

work graph to model network nodes interactions. The system stores and shares blocks with other network nodes only once it verifies incoming and outgoing pointers, sequence numbers, transaction data, and signatures.

In TrustChain, the attacker uses the subjective work graph model to increase his reputation using Sybil nodes or lowers the reputation of honest nodes by initiating the interactions with them in the network. The subjective work graph models the partial view of the interactions of the nodes in the network. The TrustChain assumes that all participant nodes contribute to other nodes in the network, and each successful contribution earns them some reputation.

The attacker creates the Sybil nodes to boost his reputation and damages the reputation of honest nodes (Fig. 4). In Fig. 4, the honest nodes *(A, B, and C)* contributing $n - units$ work to each other over a P2P network. For instance, node A contributes 4 units work to node B ($A \xrightarrow{4} B$) and receives 9 units work contributions ($A \xleftarrow{9} B$) from node B, node B contributes 8 units work to node C ($B \xrightarrow{8} C$) and receives 3 units work contributions ($B \xleftarrow{3} C$), and node C contributes

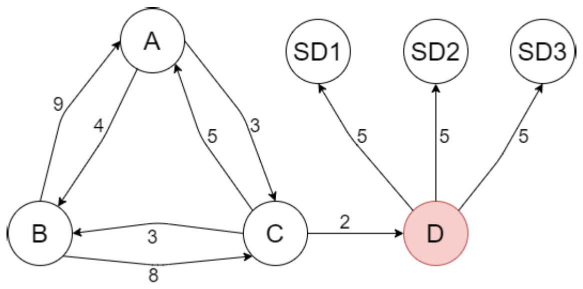

FIGURE 4. **The example of tampering nodes reputation performing by the attacker node D. The attacker creates three Sybil nodes to perform work for them that boost his contributions to get more reputation in the network.**

5 units work to node A ($C \xrightarrow{5} A$) and receives 3 units work contributions ($C \xleftarrow{3} A$). These interactions are network-wide, and nodes do not know about all such interactions. Therefore, node C has also contributed a total of 2 units work to the attacker node D ($C \xrightarrow{2} D$). The attacker node creates the three Sybil nodes *(SD1, SD2, and SD3)* and performs work for them (e.g., contributing 5 units work to each Sybil node ($D \xrightarrow{5} SD1, SD2, SD3$)) to boost his contributions to get more reputation in the network. The nodes that contribute the most are rewarded with a higher reputation score.

$$Sup = \left[\frac{\sum(X^n)}{\sum(Y^n)} \ : \ n \in \mathbb{N}, \ X \neq 0\right] \tag{1}$$

The system calculates the reputation score of the node combining Sybil nodes, and once the attacker receives more work from honest nodes in the network, the attack profitability increases. The authors [26] determine the profitability of tampering nodes reputation attack by calculating the supremum (Eq. 1). The calculation of supremum accumulates the attacker and his Sybil nodes contributions in the network.

In Eq. (1), X^n presents a sum of work the attacker node D or his Sybil nodes performed after some $n-number$ of activities and Y^n is a sum of work that obtains the attacker node D and his Sybil nodes from the network. The results in [26] show that the Sybil attack is strongly beneficial by tampering nodes reputation if supremum is infinite ($sup = \infty$). If the supremum is finite but larger than 1 ($sup \neq \infty$ & $sup > 1$) then weakly beneficial otherwise unprofitable.

TrustChain proposed an accounting mechanism-based Sybil-resistant algorithm called NetFlow [26] to limit the tampering of nodes reputation. NetFlow does not mitigate the Sybil attack, but it lowers benefits by determining and employing node trustworthiness during the consensus process. NetFlow assumes that nodes who consume resources also contribute back to the network during the transacting process. NetFlow uses a subjective work graph (Fig. 4) along with a choice set of those nodes interested in receiving some work, so the nodes get a reputation score. The Sybil nodes do not contribute to the network, and their reputation decreases over time. As a result, the system does not assign more work to node D.

4) NODES ISOLATION (PARTITION) ATTACK

In nodes isolation attack, the attacker hijacks the honest nodes and splits the network into two or more disjointed groups [21], [97]. In Fig. 5, the attacker first identifies the victim nodes, then replaces the victim nodes' peers with Sybil nodes. Next, the attacker isolates the victim nodes and disconnects their initiated transactions. The attacker nodes are now involving victim nodes in their governed blocks. In this case, the attacker gains proportional control over the system and performs various operations, such as halting transaction and block propagation, validating fake or double-spend transactions, and obtaining mining incentives.

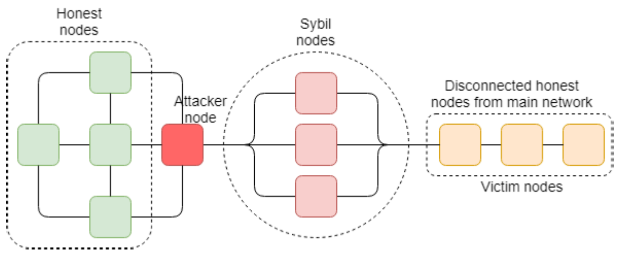

FIGURE 5. **Node isolation attack on honest nodes. The attacker divides the blockchain network into two disjoint groups. The attacker node uses the Sybil nodes and isolates the honest (victim) nodes from the network.**

If the blockchain system has insufficient computing-power, the attacker exploits this limitation [23] by using Sybil nodes [27]. For example, the blockchain network with a few participant nodes holding insufficient computing-power, the attacker comes with higher computing-power and Sybil nodes to connect with honest nodes. Using Sybil nodes, the attacker isolates the honest nodes and obstructs their communication with the main blockchain network. Once the attacker gains control on a fraction of the network, the attacker involves honest nodes in his mining process to gain more mining incentives, or affect the transaction verification. The border gateway protocol (BGP) is a routing protocol and it does not validate routing origin [47]. The attacker exploits this vulnerability by using the Sybil nodes to falsely announce some valid IP prefixes [47]. Once the honest nodes connect with Sybil nodes, the attacker can intercept the traffic and delay block propagation to affect the transaction verification. Moreover, the poor implementation of node authentication when joining the blockchain network [94]. For instance, no network joining fee, not validating IP address, or source of node connection. The attacker sees these limitations as an opportunity and builds as many Sybil nodes as he can.

There are currently no measures to mitigate the Sybil-based nodes isolation attack entirely, but there are some preventive measures to control this attack. For example, increase the computing-power in the blockchain network according to the available nodes in the network, and also monitor the nodes computing-power [23]. Monitor the round-trip time to detect irregular patterns [47], [98] during nodes data exchange and accepting transmission. Once the node identifies the irregularities, it may disconnect itself and try to randomly connect to other nodes on the network. In addition, establish a node authentication process before joining the network [94]. For instance, use network joining fee, validate the source of node connection and monitor nodes' behaviour over a certain period of time.

5) ROUTING TABLE INSERTION ATTACK

In blockchain systems, nodes keep their neighbour nodes information in the routing table (RT) and periodically update the information in RT [27]. The entry in RT is a tuple containing nodeId, network connection IP, and port. Here, the routing table insertion attack [27], [99] refers to the insertion of Sybil nodes in the RT. The attacker uses Sybil nodes to isolate honest nodes from the network and force them to insert or update the compromised node(s) in the RT. On success, the Sybil nodes will become capable of breaking the routing system, modifying or diverting message routes, block access to the transaction, affect voting outcome, and interrupt network services.

To overcome the RTI attack, the authors [27] implement a decentralised registration process to create a new node. For example, the system identifies the new node Id request and places it under the miner. The miner continuously monitors the activities of the new node and records them on the blockchain. In this process, the contributing nodes get an incentive as a respect score that enables them to access different services in the system. Use the route prediction model [99] that is based on a freenet routing algorithm.

6) SYBIL-BASED LINKING (DE-ANONYMISATION) ATTACK

Even though blockchain systems are pseudo-anonymous, it is possible to link transactions and trace a user or company behind each transaction [28]. For example, in the dusting attack [40], the attacker sends a small amount of cryptocurrency transactions to a large number of addresses. The attacker conducts a combined analysis of dusted addresses to identify the respective user when the user transfers the dusted cryptocurrency.

Blockchain system started using mixers as a service to overcome the linking attack, enhance privacy and anonymity of monetary transactions by obfuscating the transaction flow [41]. For example, the user sends the transaction to the mixer that mixes with other addresses or breaks down the transaction into smaller transactions. The mixing process muddles the original transaction connection. This mechanism was working sufficiently until *Sybil-based linking*.

Mixers operate over blockchain-based P2P systems; hence, they are susceptible to the Sybil attack. The attacker creates multiple funded addresses and joins a mixer [28]. In a mixer, the attacker creates a ring (e.g., CryptoNote mixer [42]) of his addresses and pairs them with the same users multiple times. The pairing with the same users helps the attacker to build the profile that enables him to utilise paired users transactions to de-anonymise and originate the source, thus negating the confidentiality of the user identity and transaction. The attacker uses the transaction linking information to execute phishing or cyber-extortion threats.

A Sybil-based linking attack could not be mitigated completely, but there are some techniques to restrict it [28]. For example, a non-refundable deposit to create an identity that makes it expensive for an attacker to create several identities in blockchain. Time locking on the usage of funds will require a substantial amount of funds to freeze for creating multiple identities from the attacker. The coin-age mechanism restricts the use of coins under a certain coinage. This technique takes a long time for the attacker to acquire enough aged coins to use in a mixer. If the attacker wants to participate in a mixer with lower coin-age, the system will alert broadcast to honest nodes to not join with him. The coin-age is a similar approach compared to the staking mechanism in PoS consensus to show the possession of aged coins to participate in the mixing process.

7) SYBIL-BASED DoS (DDoS) ATTACK

Despite being operating on a P2P network, blockchain is still vulnerable to DoS attacks [100], [101]. Sybil-based DoS aims to disrupt the functioning of blockchain systems or worsen the user experience by delaying the transaction or block time. The Sybil-based DoS attack can target the mixing protocol [28], and blockchain network [20].

In the mixing protocol, the attacker uses Sybil nodes to participate in the mixers where participants exchange funds to mix them [28]. In the pairing process, the Sybil nodes refuse to pair up with honest nodes. The honest nodes keep waiting to pair up with Sybil nodes, the waiting time increases the transaction mixing time that disrupts the mixer functioning.

The attack on blockchain network results in delaying the transaction or block time, exhausting the network resources, disrupting the message transmission and consensus process [20]. To trigger this attack, first, the attacker and Sybil nodes create multiple wallets on blockchain. Second, the attacker using the Sybil nodes issues numerous dust transactions (e.g., 0.0000001 ETH in each transaction) between his Sybil nodes. The blockchains by design process a limited number of transactions per block in a given time. Also, the Sybil nodes participating in the consensus process do not share their verified transactions or blocks. Therefore, the large number of transactions with small value congest the blockchain network, deny services to legitimate users, and halt the mining process. Furthermore, the Sybil-based DoS affects the mining pool performance [102] by slowing down the mining task that would discourage future users from joining the victim pool and current users might leave the pool [103].

The Sybil-based DoS attack cannot be mitigated entirely but possible to restrict it. For example, use a non-refundable deposit to create an identity that makes it expensive for the attacker to create several identities [28]. Incorporate computational constraint-based Sybil resistance techniques like Bitcoin uses PoW [28]. However, the computational constraint-based techniques would be infeasible for low computing-power blockchain systems [57]. Moreover, utilise the anti-dust model [104] to identify and prevent dust attacks. The authors add the dust transaction pool with a certain capacity of the pool. The system analyses the transaction structure based on various defined parameters (e.g., low transaction volume and fees) and adds in the dust transaction pool if dust transaction, later the dust transaction may be discarded [104].

B. DOUBLE-SPENDING RISK

Double-spending is a data consistency attack, and the attacker triggers by using various threats to gain monetary benefits in a blockchain system. Here, we discuss the threats in detail that are available in Table 8.

1) SYBIL-BASED DOUBLE-SPENDING

In Sybil-based Double-spending attack [24], the attacker uses Sybil nodes to influence communication/gossip protocol and exploits the threshold of waiting time. The attacker leverages the block propagation delay, and with 32% of computing-power he makes his fraudulent chain valid.

First, the attacker node creates several Sybil nodes and initiates a transaction A_{T0} that disseminates to other nodes on the network. Honest nodes put A_{T0} in the memory pool after verifying the transaction. Second, the attacker establishes a private chain and initiates another double-spend transaction A_{T1}, before A_{T0} adds into the block. In this stage, the attacker keeps mining his private chain and uses the Sybil nodes to delay the block propagation time on the valid chain. The Sybil nodes freeze the block propagation process by not sending new block information to honest nodes. Once the waiting time threshold exceeds, the honest nodes stop waiting and start the next round of block mining. The attacker keeps aiming to catch up with the longest chain since the growth rate of the valid chain is delayed by Sybil nodes. If the attacker succeeds to make his private chain longer, the network nodes accept the attacker's longer chain according to the longest chain rule. Now, the attacker node in a valid chain therefore double-spend transaction A_{T1} becomes valid, and the attacker gets his spent coins back.

This attack scenario is different from the 51% attack because the attacker uses Sybil nodes to delay the block propagation time. The delay supports the attacker to mine his private chain quickly by requiring only 32% of computing-power. This attack can also cause blockchain forks and waste the computing-power of honest nodes.

There exist several propositions to overcome this attack. For example, restrict miners not to mine consecutive blocks [29]. If the miner has mined a block already, then he will not execute the mining process until he receives at least one block from other miners. This change could decline the total computing-power in the blockchain network. Increase the number of confirmed blocks [24] then the attacker requires more computing resources to make the fraudulent chain longer. Furthermore, use a fee to create node identity [28] to restrict the attacker from creating multiple Sybil nodes.

2) 51% ATTACK

The 51% attack is a hash rate-based attack and the most vicious in blockchain [43], [105]. The attacker uses 51% or more computing-power in the network to successfully execute Double-spending. For example, Bob is a malicious node and forks the ledger to create his private chain. Bob produces blocks in his private chain and does not broadcast to the blockchain. Now, there are two versions of the ledger (Fig. 6), one managed by Bob and another by the honest nodes. Bob sends 50 BTC to Alice (merchant) for some product. Bob spends his bitcoins in the honest nodes chain but did not add in his private chain.

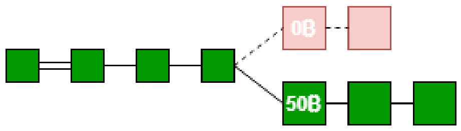

FIGURE 6. Honest nodes chain in green and fraudulent one in red.

The network nodes always adopt the longer chain. If Bob mines the blocks faster and builds a longer chain by gaining more computing-power (e.g., 51% or more), then the honest nodes will adopt the Bob chain that contains a reverse transaction (Fig. 7). In this case, Bob will get his 50 BTC back, but Alice will not retain 50 BTC sent by Bob (Fig. 8).

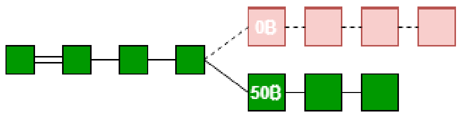

FIGURE 7. Attacker node succeeds to make the fraudulent chain longer.

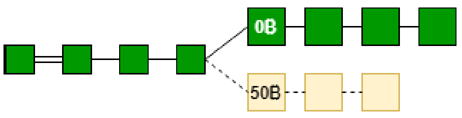

FIGURE 8. Attacker node chain is published and now it is valid one.

Hence, the attacker can successfully trigger the 51% attack by accumulating a large portion of computing-power. The attack weakens the P2P network and negates the integrity of the ledger. Also, the 51% attack could happen when the blockchain systems use an inappropriate consensus mechanism [85], [108]. For example, if the blockchain systems that contain a few participant nodes and possess insufficient computing-power [23], [37] are using the PoW consensus, the attacker comes with higher computing-power and sabotages the system. The attack results in loss of digital assets, loss the integrity of consensus and transactions.

The practicality of 51% attack is seemingly impossible on large-sized blockchain systems having high computing-power (e.g., Bitcoin and Ethereum) but possible on smaller blockchain systems holding an insufficient

TABLE 8. Framework based on SRM domain model to explore Double-spending.

Blockchain characteristic		Risk-related concept			Asset-related concept		Risk treatment concept	
Type	**Consensus**	**Threat**	**Vulnerability**	**Impact**	**System asset**	**Business asset**	**Countermeasure**	**Strategy**
Permissionless	PoW	Sybil-based Double-spending [24]	Increasing threshold of waiting time	Loss of digital assets, delay the propagation of valid blocks, blockchain forks, and waste nodes computing power	Communication / gossip protocol, Nodes identities	Block propagation (A), Digital assets (A)	Restrict to mine consecutive blocks [29]	Conceptual
							Increase confirmed blocks [24]	Detection
							Fee to create node identity [28]	Detection
Permissionless	PoW	51% attack [34], [43], [85], [86], [105], [106], [108]	Nodes can hold large portion of computing-power	Weaken the P2P network and invalid forks	Nodes, P2P Network, Ledger, Computing-power, Mining	Network reputation (I), Ledger (I)	Power monitoring tool [107]	Monitoring
							Increase confirmed blocks [43]	Detection
							Limit whale transactions [106]	Monitoring
							Transaction fee [34]	Inform
			Inappropriate consensus mechanism	Loss of digital assets, integrity of consensus and transaction	Nodes, P2P Network, Ledger, Consensus	Consensus (I), Transaction (I)	Pluggable consensus [86], [108]	Conceptual
							Delayed proof of work [105]	Conceptual
Permissionless	PoS	Long-range attack [38]	Forging timestamps	Produce blocks ahead of time	Nodes, P2P Network, Ledger	Timestamp (I), Blocks (I)	Longest chain rule [38]	Conceptual
							Context-aware transactions [38]	Detection
							Economic finality [38]	Conceptual
			Obtains the keys of past validators	Attacker generates blocks quicker than other honest nodes	Nodes, P2P Network, Ledger, Private keys, Past validators	Blocks (I), Transaction (I), Private keys (I)	Key-evolving cryptography [38]	Detection
							Moving checkpoints [38]	Inform
							Trusted execution environments [38]	Forwarding
							Economic finality [38]	Conceptual
			Stake bleeding	Stalling the main chain	Nodes, P2P Network, Ledger, Stakes, Transaction fees	Blocks (I), Transaction (I)	Longest chain rule [38]	Conceptual
							Plenitude rule [38]	Conceptual
							Economic finality [38]	Conceptual
Permissionless	PoW	Time advantage [22]	Secretly mining fraudulent blocks with time advantage	Disrupt mining process, block production, and loss of digital assets	Miners, Mining process, Computing power, Block mining time	Mining process (A), Ledger (I), Digital assets (A)	Limit time advantage [22]	Monitoring
							Monitor time differences [22]	Monitoring
							Increase confirmed blocks [22]	Forwarding
Permissionless	PoW	Eclipse-based Double-spending [30], [46]	Accepting unconfirmed transaction	Disrupt fast payment mechanism, and loss of digital assets	Transaction verification, Block confirmations, Ledger	Fast payment (I, A), Digital assets (I)	Increase confirmed blocks [109]	Detection
			Node connects directly to incoming connections	Restrict node to learn the rest of the blockchain network	Nodes, IP addresses, Node connection, Transaction	Communicating/ gossiping (A), Transaction verification (I)	Disable incoming connections [30], [110]	Inform
							White-listed nodes [30]	Forwarding
							Random outgoing connections [30], [110]	Conceptual
							Deterministic random eviction [30]	Detection
							Feeler connections [30]	Inform
							Anchor connections [30]	Inform
Permissionless	PoW	Border gateway protocol hijacking [47], [111]	Possible to hijack IP prefixes by corrupting routing table	Reroute traffic by hijacking IP prefixes and obstruct the block propagation, introduce block races, waste mining pool computing power, blockchain forks, selfish mining or decrease miners revenue	Border gateway protocol, IP prefixes, Network traffic, ISP, Routing table	Block propagation (A), Routing table (I), Transaction (I)	Increase nodes connection diversity [47]	Conceptual
							Multi-homing of mining pool [47]	Detection
							Random outgoing connections [47]	Forwarding
							Monitor round-trip time [47]	Monitoring
							Anomaly detection mechanisms [47]	Detection
							Hosting several gateway [47]	Broadcasting
							Randomly request a block [47]	Conceptual
							Increase confirmed blocks [111]	Detection
							Selectively choosing peers [111]	Conceptual
							Encrypt Bitcoin communication [47]	Detection
							Use randomised TCP port [47]	Inform
							Periodically send UDP messages [47]	Inform
Permissionless	PoW	0-confirmations race attack [21], [31], [48], [49], [112]–[114], [120]	Accepting unconfirmed transaction	Disrupt fast payment mechanism, and loss of digital assets	Transaction, Transaction verification, Block confirmations, Ledger	Fast payment (I, A), Digital assets (I)	Increase confirmed blocks [48], [49]	Detection
							Closed-form formula probability [112]	Conceptual
							Enhance network policy [17], [113]	Inform
							Listening period [31]	Conceptual
							Insert observer [31], [114]	Conceptual
							Alerting honest nodes [31]	Broadcasting
							Forward double-spend transaction [120]	Forwarding
							E-cash protocol [117]	Detection
							Enhanced observers [119]	Monitoring
Permissionless	PoW	Finney attack [17], [21], [30], [114]–[116]	Accepting 1-confirmation transaction	Disrupt fast payment mechanism, and loss of digital assets	Nodes, Transaction, Transaction verification, Block confirmations, Ledger	Fast payment (I, A), Digital assets (I)	Increase confirmed blocks [21], [116]	Detection
							Logarithmic waiting time [115]	Conceptual
							Gambler's ruin problem [17], [118]	Inform
							Listening period [31]	Conceptual
							Insert observer [31], [114]	Conceptual
							Alerting honest nodes [31]	Broadcasting
Permissionless	PoW	Vector76 attack [21], [30], [49], [50], [114]	Accepting 1-confirmation transaction	Disrupt fast payment mechanism, and loss of digital assets	Nodes, Transaction, Transaction verification, Block confirmations, Ledger	Fast payment (I, A), Digital assets (I)	Increase confirmed blocks [21], [49]	Detection
							Listening period [31]	Conceptual
							Insert observer [31], [114]	Conceptual
							Alerting honest nodes [31]	Broadcasting
			Possible to have multiple full nodes	Loss of digital assets	Nodes, Transaction, Block confirmations, Ledger, Exchanges, E-wallets, Static IP address	Digital assets (I)	Disable incoming connections [50]	Inform
							Well-connected inbound connections [50]	Detection
							Monitor outgoing node connections [50]	Monitoring

amount of computing-power [23]. Blockchain systems that have recently suffered from 51% attack include Ethereum Classic (ETC), Feathercoin (FTC), Bitcoin Gold (BTG), Vertcoin (VTC), and Verge (XVG) [23], [37]. At the time of the attack, all of these blockchain systems were holding a limited number of nodes, insufficient computing-power, and using PoW consensus. Double-spending is the one result of a 51% attack. The attacker can achieve selfish-mining, prevent new transactions from gaining confirmations, and blockchain forks.

To protect against 51% attack, implement a power monitoring tool to continuously monitor computing-power of nodes and restrict when reaching a certain amount of computing-power [107]. M. Rosenfeld [43] states that by increasing the number of confirmed blocks (it means, waiting for more confirmations before accepting the transaction e.g., 6 confirmations are required in Bitcoin) would decrease the success probability of hash rate-based attack. The PoW-based blockchain system requires a mechanism to limit the number of whale transactions, reduce the size of transactions fee, and prevent honest miners colluding to mine a whale block for more profitable mining [106]. Incorporate transaction fee [34] as an incentive to keep nodes honest in a blockchain system. Use a pluggable consensus mechanism [86], [108] to facilitate consensus diversity based on the business models and requirements of the blockchain system. Komodo presents delayed proof of work (DPoW) [105] for unspent transaction output-based blockchain systems (e.g., Bitcoin) to add an impenetrable security layer for mitigating 51% attack. DPoW combines the notary node network and recycles the hash rate. The notary node network stores backups of individual blocks on multiple blockchain networks (called notarised blocks). Once notarisation is complete the history of every chain using dPoW becomes immutable.

3) PoS LONG-RANGE ATTACK

The PoS long-range attack [38] is similar to 51% attack. Currently, blockchain systems (e.g., Ethereum) attempt to switch over PoS to overcome PoW shortcomings [38]. In PoS long-range attack, the attacker creates a private chain starting from the genesis block and overtaking the main chain. The attacker forges the blocks and adds double-spend transactions in his private chain. The attack succeeds once the attacker's private chain becomes longer compared to the main chain. There are three types of PoS long-range attacks: simple long-range, posterior corruption, and stake bleeding.

In simple long-range attack, the attacker forges timestamps to produce blocks ahead of time to advance his private chain and overtake the main-chain (Fig. 9). In the implementation of PoS protocol, where the nodes do not verify the block timestamps, the attacker can exploit the block timestamps according to his will. As a result, both branches become valid. Once the attacker branch is ahead of the main branch, then other nodes will accept the longer chain.

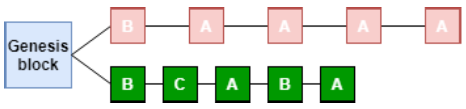

FIGURE 9. To compute blocks ahead of time and try to overtake the main chain, the attacker forges the timestamps.

In posterior corruption, the attacker uses private keys of such nodes (e.g., Node B) that leave the validating process and cash out the stakes. For example, *NodeB* retires and removes his stakes after validating the first $n - blocks$ in the main chain. Now, *NodeB* is not a part of the block validation process and cannot sign new blocks. However, *NodeB* can still sign the first $n - blocks$ of any branch of that blockchain [38] using his private key. There are two possible ways the attacker can get access to the *NodeB* private key. Firstly, after leaving the validation process the security of his private keys is not a priority and the attacker somehow gets access to it. Secondly, the attacker can convince *NodeB* to participate with him to launch the long-range attack. *NodeB* has already left the validation process and nothing at stake. Hence, there are likely high chances to perform this attack by teaming with the attacker. In this case, the attacker can sign valid blocks and increase his chances of generating blocks quicker than other honest nodes (Fig. 10).

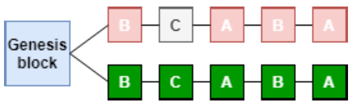

FIGURE 10. To make the private chain (upper chain) more competitive, Node B joins the attacker and attempts to conquer the main chain.

In stake bleeding (Fig. 11), the attacker starts stalling the main chain and gets more time to create blocks in his private chain [38]. For instance, when the attacker becomes a slot leader, he skips the chance to stretch the main chain, and no new block will be added to this slot. The attacker earns no incentives, and his stakes will decrease (or bleed) in the main chain. However, he keeps publishing blocks in his private chain and raises his stakes through transaction fees. The attacker's private chain grows faster than the main chain from this point on and gradually becomes longer. The attacker adds one transaction to redistribute the stakes to other validators just before the forged private chain is released, which would not violate the "honest majority" presumption. Therefore the attacker private chain will become valid.

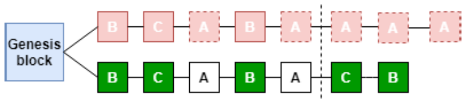

FIGURE 11. Attacker stakes in his private chain (upper chain) steadily increase and are more often chosen as a block validator while he tends to lose stake in the main chain and is thus less frequently chosen as a block validator in the main chain.

Various techniques are available to overcome the PoS long-range attack [38]. Individually, these techniques can not completely mitigate the attack but can restrict to a certain level by combining these solutions [38].

In PoS, due to weak subjectivity, when the nodes come online they cannot determine which node is longer/valid because having many different branches on the blockchain. Therefore, the nodes may be tricked into accepting the attacker branch and participate in his block validation process. Like PoW, the PoS longest chain rule can overcome weak subjectivity issues, and the newly added nodes will always join the longer chain.

In context-aware transactions, universal hash time is utilised to establish transaction references bound to specific points in time. This technique restricts the attacker to copy valid transactions from the main chain to his private chain. If the attacker copies the valid transactions from the main chain then honest nodes will reject considering inconsistency in the attacker private chain.

The economic finality approach ensures that the validators participating in the validation process have something to lose once they misbehave. This technique does not eradicate the attack but, by financial penalty, discourages the attacker.

The key-evolving cryptography technique makes used keys immediately useless. The technique forcefully creates and assigns a new private key to the validator for participating in the next block validation round. For example, in posterior corruption, if Node B wants to sign old blocks then he will be unable to use the already used private key.

Moving the checkpoints technique enables a quota that allows only $n-number$ of blocks to be reorganised. The quota changes according to the protocol settings and configurations. For example, in Peercoin [37] the quota is limited to one month of blocks and the NXT coin [38] quota is just for a few days or hours.

Use trusted execution environments (TEE), for example, Intel software guard extensions (SGX). The SGX allows to perform signing within a trusted domain and signing private keys are not available beyond the TEE.

The plenitude rule aims to calculate a branch's block density from the moment the branch is created. Assuming that a minority is always the attacker, then the main branch will always be denser than the opposing branches. It is reasonably easy to define the main chain and defeat weak subjectivity by following this law. However, if more than 34% of the validators are malicious, the system fails.

4) TIME ADVANTAGE

This attack scenario is similar to 51% attack, the only difference is that the attacker uses time advantage [22] over honest miners along with hash rate to produce fraudulent blocks and perform double-spending. Time denotes the average time in seconds required for the whole network (e.g., including both honest and attacker nodes) to mine a block. The time advantage, assuming that the attacker has been mining fraudulent blocks secretly with a time advantage of $n - seconds$ over honest miner nodes. The attacker interrupts the mining process, affects the block propagation, and steals the digital assets upon a successful attack.

The attack scenario of time advantage is an extension of the hash rate-based double-spending attack model by S. Nakamoto [44] and M. Rosenfeld [43]. The extended attack version uses two different approaches: the time-based generalised model and the time-based ledger state model. The time-based generalised model adds a time parameter to the mining process, enabling an attacker to secretly mine blocks with enough time advantage to achieve a double-spending. In contrast, the time-based ledger state model considers the times at which the honest and attacker nodes last mined a block. This model represents the states in terms of the length of the chains where the attacker and the honest nodes are mining, and they can be different. The comparison shows [22] that the probability of carrying double-spending with time advantage is coinciding.

The authors [22] calculate the probability of a double-spending using time advantage, number of confirmations, and computing-power. Along with time advantage, if the number of confirmations decreases or the attacker computing-power increases, the double-spend attack is doable. For example, if an attacker gains control of 40% of the network's computational power, the double-spending is almost impossible to contain. This situation is very unlikely in high computing-power blockchain systems (e.g., Bitcoin and Ethereum) but possible in low computing-power blockchain systems.

To mitigate time advantage attack, the authors [22] recommended to monitor time differences after the new block is produced, and implement a mechanism to limit the time advantage when the honest miner and the attacker last mined a block. In addition, increase the number of block confirmations [22], [43] to make it expensive for the attacker to carry this attack.

Blockchain reorganisation: Sybil-based Double-spending, 51%, PoS long-range, and time advantage attacks are categorised under the blockchain reorganisation (also known as alternative history) attack. In these attacks, the attacker creates a private chain that includes double-spending transactions, and relies on his computing-power (and stakes in PoS) to find blocks quicker than honest nodes. If the attacker succeeds to make his private chain longer then according to the longest chain rule the nodes adopt the longest chain of blocks. Otherwise, the attacker has wasted his resources. Blockchain reorganisation can also happen due to an accidental fork, when two or more miners (honest nodes) find the next block at approximately the same time and compete for their own chain validity. The nodes with more computing-power add new blocks faster and make a longer chain. The blocks of the losing chain marked as orphaned blocks. In accidental forks, there is no double-spending but the losing chain nodes (e.g., miners and mining pool) lose their work and incentives [89].

5) ECLIPSE-BASED DOUBLE-SPENDING

The Sybil attack is a network-wide attack; in contrast, an eclipse attack only targets a particular node(s) [24]. The attacker floods the victim node with his IP addresses to disengage the victim node from the blockchain network and directly attached himself to the victim node. The attacker prevents the victim node from learning the rest of the blockchain network by not communicating/gossiping with other peer nodes. The eclipse attacks are performed on a specific node(s) such as prominent miners or merchants. The studies [30], [46] explain the possibility of Eclipse-based Double-spending on the blockchain systems. In Eclipse-based Double-spending, the attacker exploits 0-confirmations or N-confirmations in fast payments to trigger double-spend.

To make fast payments desirable for both the merchant and customer, the 0-confirmations approach is followed [31]. The 0-confirmations transaction (also known as an unconfirmed transaction) stores in mempool of the honest nodes and not yet included in the blockchain. The attacker exploits it and refers 0-confirmations transaction to a merchant that releases goods by assuming that confirmations will occur. In 0-confirmations double-spending, the attacker eclipsed the merchant node (Fig. 12) that accept a transaction A_{T0} with no confirmation. Next, the attacker creates a new transaction A_{T1} for double-spend and broadcasts to the rest of the network. The merchant releases the goods to the attacker, and till this stage merchant does not know that his connections are eclipsed and cannot tell the blockchain network about A_{T0}, later blockchain network confirms A_{T1} as valid and discards A_{T0}. Hence, the attacker gets his goods without paying.

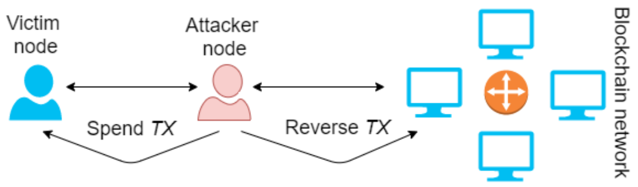

FIGURE 12. Eclipse-based 0-confirmations Double-spending.

In N-confirmations double-spending, the attacker eclipses the $n - fraction$ of miners along with the merchant (Fig. 13). The eclipsed merchant waits until block depth $N - 1$ confirmations before releasing the goods. The attacker shows the confirmations to a merchant from eclipsed miners. The merchant is convinced and sends the goods to the attacker. Upon completion of goods purchase, the attacker sends the actual blockchain view that makes eclipsed miners blockchain orphan. Thus the attacker procures goods without paying.

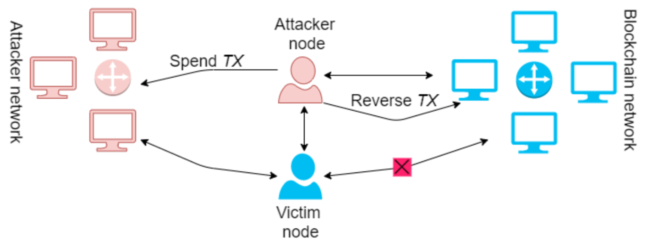

FIGURE 13. Eclipse-based N-confirmations Double-spending.

To overcome Eclipse-based Double-spending risk, the study [109] asserts to wait for more block confirmations (e.g., 10 block confirmations to lessen the probability of this attack). However, waiting for more blocks confirmations will make the payment slow between merchant and customer, but the payment will be safe. The authors [30], [110] apply the combination of different techniques to control Eclipse-based Double-spending. For example, disable the direct incoming connections. Use the white-listed nodes to choose outgoing connections, for instance, well-connected peers/miners. Random selection of addresses when making outgoing connections. The deterministic random eviction approach follows the *new* and *tried* addresses tables. The system deterministically hashes each address to a single bucket and assigns a single slot in a *new* table. If the connection succeeds, then the address moves from *new* to *tried* table. This approach reduces the attack addresses that attacker use for making outgoing connections. The feeler connections in which the nodes establish short-lived test connections to randomly-selected addresses and if connection succeeds then the address is inserted into white-listed nodes. In anchor connections, integrate an anchor table to record the address of current outgoing connections. Once the node restarts and tries to make outgoing connections again, the system selects old addresses from an anchor table and two additional connections to persist rotation rate.

6) BORDER GATEWAY PROTOCOL HIJACKING

BGP is a routing protocol to exchange network routing information between independently operated networks such as internet service providers (ISPs) or autonomous systems (AS) [20], [47] (e.g., SC-based dApp). The BGP hijacking is a routing attack and the attacker influences the ISPs/AS to make false announcements over the routing system to divert traffic. The attacker hijacks IP prefixes to partition the blockchain or obstruct the block propagation to perform double-spending. Monthly, an average of 100 BGP exploitation occurs [23] over a Bitcoin blockchain. The attacker can isolate up to 50% hash rate on the Bitcoin network by hijacking fewer than 100 BGP IP prefixes [20].

The attacker uses BGP hijacking and slows down the propagation of new blocks to perform Double-spending. The attacker exploits block propagation delay to render 0− or $N - confirmations$ double-spending. Moreover, the attacker can engineer block races, waste mining pool computing-power, blockchain forks, selfish mining attacks, or decrease miners revenue.

The study [47] discusses a few preventive measures against BGP hijacking attack. For example, increase the diversity of node connections, it means multi-homing of the mining pool on the blockchain via multiple and distinct paths. A single-homed node can use a virtual private network (VPN) service to create multiple and distinct paths. Connect with random peers while making outgoing connections to avoid biased decisions. During BGP hijacking, round-trip time (RTT) increases; thus, by monitoring RTT, a node can quickly detect the attack and add extra random connections to protect against it. Deploy anomaly detection mechanisms and monitor sudden changes, such as the distribution of connections, the time elapsed between request and answer, and simultaneous disconnections of peers. Once an outgoing connection fails, the network refreshes its connections to embrace intentional or attacker-based connection churn. Hosting several gateways in different AS would make blockchain more robust to routing attacks. The nodes should randomly request a block from multiple connections, so the attacker does not keep waiting for the node to deliver a block.

In Ethereum platform, to minimise the risks of BGP hijacking-based Double-spending, the authors [111] increase the number of blocks confirmations needed before transaction finality. Also, selectively choosing peers for querying transaction status, where a merchant verifies whether an issued transaction is committed or not before completing the purchase. Moreover, there are a few countermeasures [47] specifically for the Bitcoin blockchain. For example, encrypt Bitcoin communication or use message authentication code to validate that the communication message content has not changed. Use distinct control and data channels, it means, use a randomised transmission control protocol (TCP) port rather than relying on the default port (8333) to communicate with other nodes on the network. Periodically send UDP messages that enable a node to realise that he is out-of-sync and establish new connections.

7) 0-CONFIRMATIONS RACE ATTACK

0-Confirmations race attack targets fast payments transaction when a merchant accepts payments immediately with 0-confirmations [21], [31]. The attacker sends two conflicting transactions in the blockchain with the same inputs but different transaction fees. One transaction (A_{T0}) to the merchant directly to pay for goods and the second transaction (A_{T1}) to himself with a higher transaction fee. Here, the attacker tries to exploit the intermediate time requires for initiation and confirmation of two transactions. Once, the A_{T1} will be mined and accepted because of the higher transaction fee by the nodes, then A_{T0} becomes invalid.

The 0-confirmations race attack is different from the Eclipse-based and BGP hijacking-based 0-confirmations attack because in 0-confirmations race attack, the attacker indulges in a race to make his double-spend transaction valid by exploiting the intermediate time between two conflicting transactions and using a higher transaction fee.

The authors [48] suggested to wait for at least one block confirmation before sending out the goods, but this approach could lead to Finney or vector76 attacks [17], [21]. The merchant should wait for a few more confirmations, like coinbase crypto exchange requires only 3 confirmations to mark bitcoin transaction final [49]. The waiting time for block confirmations diminishes the acceptability of fast payments. The study [112] presents a closed-form formula (combining transaction validation time) to calculate the probability of double-spending in race attack. In addition, enhance network policy [17], [113] to guide about how to set a block confirmation number considering the value of the transaction.

The authors [31] present three techniques to detect 0-confirmations race attack. The listening period (approx. 3.354 seconds) helps the merchant to monitor transaction legitimacy before sending the goods. Inserting observer is an extra layer on a listening period where the merchant inserts an observer that directly relays to him. The observer detects a double-spend transaction in a few seconds and informs the merchant about this attempt. Alerting the vendor on the blockchain once other honest nodes on the network receive a double-spend transaction. Similarly, the Grundmann *et al.* [120] advice to forward double-spend transaction to warn the peers in the network. Furthermore, the authors [114] insert the observers in the transaction pool to detect if another transaction with the same inputs, E-cash protocol [117] incorporates a large group of trustworthy hosts as observers to detect double-spending in real-time, and the enhanced observers (ENHOBS) [119] combines the listening period and observers to monitor the double-spending transaction.

8) FINNEY ATTACK

Finney attack is a 1-confirmation pre-mining attack [17], [30]. Similar to the 0-confirmations race attack, the attacker utilises two conflicting transactions with the same inputs [21]. Firstly, the attacker per-mined a block with a transaction to himself (A_{T0}) and does not broadcast the block. Secondly, the attacker sends a second transaction (A_{T1}) with the same input to the merchant directly to pay for goods. Once, merchant accepts the transaction and sends the goods to an attacker, then the attacker releases his privately pre-mind block into the blockchain. Now, the attacker A_{T0} will take precedence over merchant A_{T1} thus the attacker would receive his coins back, and merchant loses his payment and product. However, the Finney attack can fail and the attacker loses his pre-mined block reward. For example, the probability is t/T that another block will be mined earlier than the attacker block [45]. Here, t is the time from finding the block until the attacker sends a payment and the merchant accepts, and T is an average time to find a block. The Finney attack is not a hash rate-based attack but slightly relies on hash rate, if the attacker has a low hash rate then it is less likely he would carry out this attack.

The possible countermeasure is to wait for a few more block confirmations [21], [116] before completing the transaction. The authors [115] present the logarithmic waiting time for large transactions. For instance, the recipients of large transactions are advised to wait a time logarithmic in the chain's length. The gambler's ruin problem [17], [118] simulates the probability of the Finney-based double-spending. The results show the probability increases with the attacker mining power. The techniques (listening period, insert observer, and alerting nodes) [31] presented for 0-confirmations race attack can help to limit the Finney attack.

9) Vector76 ATTACK

Vector76 attack is also a 1-confirmation attack that combines 0-confirmations race and Finney attacks [21] to disrupt the fast payments. The attacker targets the exchanges and e-wallets that connect with a static IP address and accepts direct incoming connections. The attacker withdraws coins immediately when exchanges/e-wallets accept the transaction with 1-confirmation.

To launch a vector76 attack, the attacker maintains two full nodes (e.g., Node A and Node B), Node A is directly connected with exchange/e-wallet and Node B with the rest of the blockchain network. The attacker creates two conflicting transactions where one transaction (A_{T0}) on Node A (connected with exchange/e-wallet) and second transaction (A_{T1})

to on Node B (connected with other nodes on a blockchain). So far, both transactions have not been broadcast to the blockchain. The attacker starts mining the block with A_{T0} on Node A. Once the attacker gets a block solved then instead of releasing the block publicly the attacker convinces exchange/e-wallet with 1-confirmation and at the same time release A_{T1} on Node B. The A_{T0} is known only to the attacker and connected exchange/e-wallet that will deposit the transaction into the attacker account after 1-confirmation (because of a block on Node A). The attacker would immediately withdraw the coins from his account and later a well connected Node B transaction will become valid thus double-spending attack is successfully carried. The attacker also pays the price (e.g., 1 Block that he mined on Node A) to execute this attack, but the reward is much higher than the cost.

To protect against vector76 attack, the node should avoid 1-confirmation transaction [49]. The countermeasures (listening period, insert observer, and alerting nodes) [31] presented for the 0-confirmations race and Finney attack [21], [116] can limit the Vector76 attack. In addition, repudiate inbound connections or define inbound connections from well-connected nodes [50]. Monitor outgoing connections to detect false information from the attacker (e.g., double-spend transaction and blockchain connection).

IV. EXAMPLE OF FRAMEWORK USE

In this section, we discuss the use of a newly developed framework. We selected two healthcare dApps (MedRec and MIStore) that are built upon the Ethereum platform.

A. HEALTHCARE dApps

Healthcare data is sensitive, and it must be integral. Falsified or misplaced health records can cause major issues during the patient treatment process [90]. In previous years, several research studies have been conducted to preserve patients medical health data using blockchain to ensure data integrity [52], patient ownership to his data [32], easy exchange of medical data [51], and medical insurance claims [33].

The **MedRec** [32] dApp is a decentralised electronic health record (EHR) and medical research data management system (Fig. 14). The dApp enables patients to control their data and access their medical information across providers and treatment sites. It utilises the blockchain to achieve fast access to medical data, system interoperability, and improved data quality.

MedRec includes hospitals, patients, and researchers (e.g., doctors, institutes, and public health authorities) as network participant nodes. Blockchain node combines client node, smart contracts, mining, mining protocol, and immutable ledger. In MedRec, each participant node manages one client node, where an Ethereum client (e.g., PyEthApp) implements the Ethereum platform specification (e.g., connection with P2P network, encoding and sending transactions). MedRec service monitors the real-time changes to signal EHR manager that issues a patient notification and

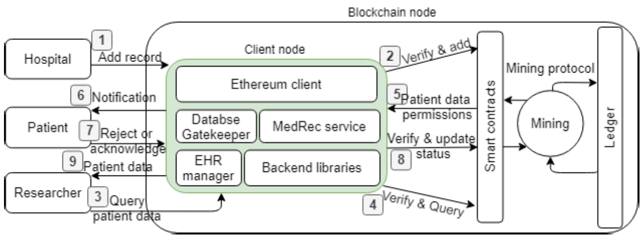

FIGURE 14. MedRec system orchestration.

syncs the off-chain database using a database gatekeeper. Backend libraries communicate with an Ethereum client to facilitate the MedRec service. The hospitals are responsible for adding the patient medical health record. The working procedure of MedRec dApp (Fig. 14) is discussed in detail in Table 9.

The **MIStore** [33] dApp is a decentralised medical insurance storage system (Fig. 15). MIStore utilises blockchain ledger immutability property to provide high-credibility insurance services to users.

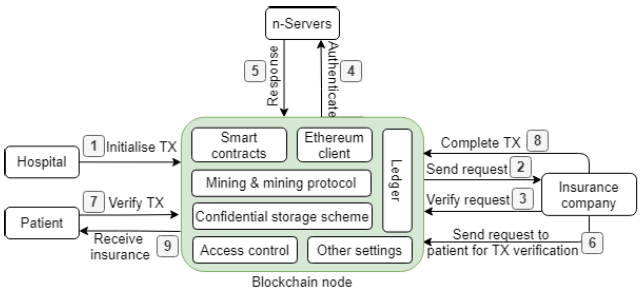

FIGURE 15. MIStore system orchestration.

MIStore includes hospitals, patients, the insurance company, and n-servers as network participant nodes. N-servers are nodes that verify the authenticity of a transaction. Blockchain node combines Ethereum client to implement the Ethereum platform specification, smart contracts, mining, mining protocol, and immutable ledger. Confidential data storage scheme enables confidentiality for patients medical data, access control manages operations of the insurance company and other nodes accessing patients medical data, and other settings facilitate the operations of MIStore. The hospitals are responsible for adding the medical treatment costs of patients that initialise the insurance claim transaction. The working procedure of MIStore dApp (Fig. 15) is discussed in detail in Table 9.

The working procedure of both dApps categorised into 4 stages (initialisation, verification, processing, and response) to execute a transaction (Table 9). The table also shows the components of both dApps. We utilise Table 9 and identify the commonalities in MedRec and MIStroe dApps to build an architecture (Fig. 16) that characterises the system assets of both dApps at different layers. For example, the *application layer* provides an assertion to system users for accessing

TABLE 9. Working procedure of MedRec and MIStore, and components.

	MedRec	MIStore
Initialisation	*1. Add record* - Hospital initiates add record transaction *3. Query patient data* - Researcher query patient data for medical-related research *7. Reject or acknowledge* - Patient can reject or acknowledge the data request	*1. Initialise TX* - Hospital initiates the insurance claim on behalf of patient *2. Send request* - System forwards request to insurance company *6. Request for TX verification* - Insurance company sends request to patient for verification of the insurance claim transaction
Verification	*2. Verify and add record* - System verifies the add record request from hospital *4. Verify and query* - System verifies the researcher request for patient data *8. Verify and update status* - System verifies the patient response on researcher data request	*3. Verify request* - Insurance company asks to verify TX request *4. Authentication and Verification* - n-Servers authenticate and verify the request *7. Verify TX* - Patient verifies the TX request
Processing	*5. Check data permissions* - System check the permission settings if already assigned to data for sharing *10. Incentives* - Researcher and hospital get incentives as patient medical data	*8. Complete TX* - System completes the insurance claim TX request initiated by hospital on behalf of a patient *10. Incentives* - Hospital and n-Servers get incentives
Response	*6. Notification* - System notifies the patient about his decision on reject or acknowledge the data request *9. Patient data* - Researcher gets patient data	*5. Response* - n-Servers send response to system after verifying and authenticating the TX request *9. Receive insurance* - Patient receive insurance
Components	Transaction verification, P2P network, cryptography, data validators & miners, ledger, transactions, Ethereum virtual machine (EVM), Ethereum client, consensus, mining protocol, smart contracts, wallet, digital assets, data hash id, *EHR manager, DB Gatekeeper, backend libraries*	Transaction verification, P2P network, cryptography, data validators & miners, ledger, transactions, Ethereum virtual machine (EVM), Ethereum client, consensus, mining protocol, smart contracts, wallet, digital assets, data hash id, *n-Servers, access control, confidential storage scheme*

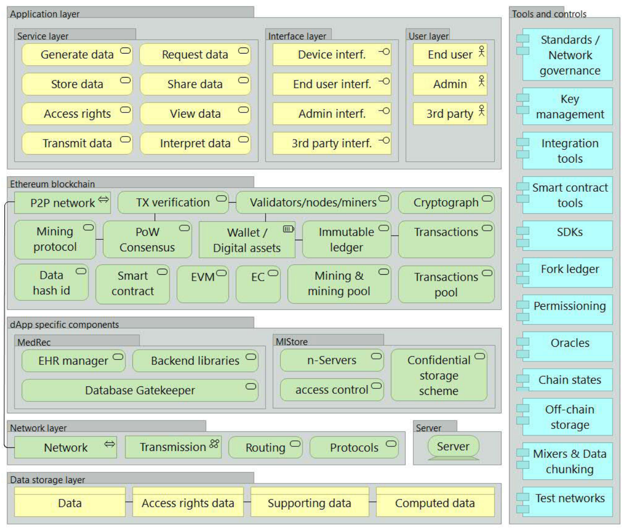

FIGURE 16. Components of Ethereum-based MedRec and MIStore dApps.

different resources. The *user layer* exposes the users who interact with the application. The *interface layer* presents the various interfaces of the application. The user interacts with the services, which are available in the *service layer*. The *Ethereum blockchain* and *Controls* layers show the different components and tools associated with the Ethereum platform. The *dApp specific components layer* presents the additional settings in both dApps. The *network layer* is responsible for transmitting communication using different protocols and routing when interacting with external systems and databases. The *server layer* hosts off-chain services and resources. The *data storage layer* shows the database as off-chain storage.

B. RISK MODEL

We evaluate the Sybil attack in MedRec and Double-spending in MIStore (Table 10) by following the framework (Table 7 & 8). We identify the security threats belonging to the Sybil attack in MedRec dApp, and security threats that could trigger Double-spending in MIStore dApp. Also, to alleviate the identified security threats, we accumulate countermeasures to integrate into both dApps.

In MedRec, the nodes are limited, and the attacker can create Sybil nodes to trigger nodes isolation attack. Using nodes isolation attack, the attacker uses honest nodes to participate in the attacker governed blocks or consensus process. The results of this attack can lead the attacker to validate wrong data (e.g., to register an invalid patient), steal and sell the patient medical data, or verify illegal drug prescriptions. Furthermore, in MedRec, the attacker can perform a Sybil-based DoS attack to halt the services of MedRec.

In MIStore, Double-spending can be used for insurance frauds. In Eclipse-based Double-spending, the attacker will eclipse the node of a patient to perform Double-spending. The hospital initiates the insurance claim transaction (I_{T0}) and sends it to an insurance company (IC). IC verifies the legitimacy of I_{T0} and sends I_{T0} to a patient for verification. In this stage, the attacker receives I_{T0} transaction and sends conflicting transaction (A_{T1}) to a patient for verification. The patient verifies the A_{T1} and sends a response message to the IC. Here, IC only knows I_{T0}, therefore completes the I_{T0} transaction. Later, the system invalidates the conflicting transaction (A_{T1}), and the attacker gets insurance. Also, IC validating transaction with 0-confirmations, hence, suspect to double-spending due to 0-confirmations race attack.

In Fig. 17, an abstraction of Sybil and Double-spending risks are presented. The diagram illustrates both security risks, their associated threats, vulnerabilities, and affected assets. For example, the attacker commands the Sybil or Double-spending risk using different threats by exploiting different vulnerabilities, which provoke the negative impact and negate the security criteria of the business assets. The vulnerabilities are connected to the system assets and they depict their weaknesses that enable the attacker to harm valuable assets.

The diagram (Fig. 17) shows that the attacker can execute nodes isolation and Sybil-based DoS in MedRec. Nodes isolation attack has three different vulnerabilities. For example, insufficient computation power (V#1) in the network affects the P2P network and PoW consensus. The BGP does not validate routing origin (V#2) that affects the transaction verification. Also, the dApp has a limited number of

TABLE 10. SRM of MedRec and MIStore dApps.

dApp	Risk	Threat	ID	Vulnerability	Affected asset	Countermeasure
MedRec	Sybil attack	Node isolation attack	**V#1**	Lack of computation power	P2P network, PoW consensus	Increase computing power
			V#2	BGP does not validate routing origin	Transaction verification	Monitor round-trip time
			V#3	No proper authentication of nodes	Transaction verification, Validator/ nodes/ miners	Network joining fee
						Validating node connection
						Monitoring activities of node
		Sybil-based DoS	**V#4**	Sybil nodes can participate in mining process	Mining, Mining pool	Use computational constraint-based techniques
MIStore	Double-spending	Eclipse-based Double-spending	**V#5**	System connects node to incoming connections	Transaction verification, Validator/ nodes/ miners, Mining, Mining pool, Ledger, Digital assets	Disable incoming connections
						White-list nodes for outgoing connections
						Random selection of new connection
			V#6	Accepting unconfirmed transactions	Transactions, Transactions pool, Ledger, Digital assets	Increase confirmed blocks
		0-confirmations race attack	**V#6**	Accepting unconfirmed transactions	Transactions, Transactions pool, Ledger, Digital assets	Increase confirmed blocks
						Listening period
						Insert observer
						Alerting honest nodes
						E-cash protocol
						Enhanced observers

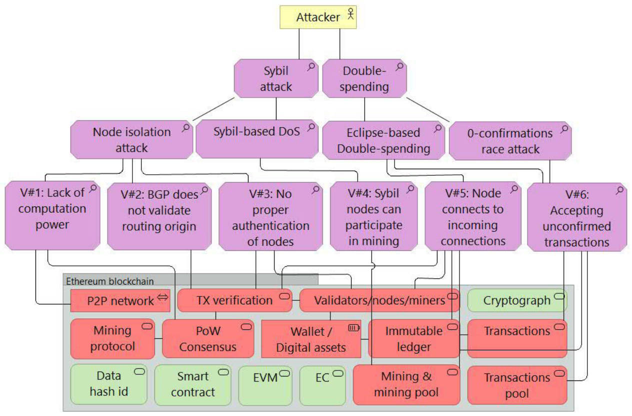

FIGURE 17. Risks architecture of MedRec and MIStore dApps.

nodes controlling insufficient computing-power, therefore, it is more vulnerable to this attack. Improper authentication of nodes (V#3) make the transaction verification, validators/nodes/miners vulnerable. In Sybil-based DoS attack, the Sybil nodes can participate in mining (V#4), thus halting the mining and mining pools operations.

In MIStore, the attacker can trigger Eclipse-based Double-spending and 0-confirmations race attack. The diagram (Fig. 17) shows the system connects the node to incoming connections (V#5) without checking the attacker IP addresses flooding the victim node connections. V#5 affects transaction verification, validators/nodes/miners, mining, mining pools, ledger, and digital assets. Also, accepting unconfirmed transactions (V#6) affects the transactions, transaction pool, ledger, and digital assets.

C. COUNTERMEASURES

The countermeasures (Table 10) are set based on the framework (Table 7 & 8) to mitigate Sybil and Double-spending risks vulnerabilities within MedRec and MIStore dApps.

For instance, by adding more computing-power, V#1 could be handled. To overcome V#2, monitor the round-trip time to detect irregular patterns. Vulnerability V#3 is restrained by performing node authentication before joining the blockchain network, such as asking network joining fee, validating node connection, and monitoring nodes activities. V#4 is controlled by managing the computing-power, for example, using a computational constraint-based Sybil resistance technique that requires spending computational resources proportional to the number of identities produced. Vulnerability V#5 can be controlled by disabling incoming connections, introducing white-listed nodes for making outgoing connections, and using only random selection for establishing new connections. In Eclipse-based Double-spending to minimise V#6, increase confirmed blocks. In 0-confirmations race attack, the V#6 could be mitigated by using more confirmed blocks, adding a listening period, inserting observer, alerting honest nodes, E-cash protocol, and enhanced observers.

V. EMERGING CHALLENGES OF BLOCKCHAIN

Currently, blockchain is facing several other security and implementation challenges that are hindering the acceptance of blockchain systems on a wide-range.

A. OTHER SECURITY CHALLENGES

In year 2020, the attackers carried 122 attack events and stole around $3.8 billion (Table 11) [53]. Ethereum (by its cryptocurrency Ether) is a second-largest and state of the art blockchain platform for building dApps. The attackers mostly targeted Ethereum dApps and carried 47 attacks that cost $436.36 million. In this section, we present an overview of the various security challenges of blockchain systems.

TABLE 11. Blockchain-related attack events in 2020 [53].

Target	Money lost	# of attacks
Wallets	$3,027,108,209	27
Ethereum dApps	$436,364,097	47
Exchanges	$300,149,169	28
Tron dApps	$10,000,189	3
EoS dApps	$2,848,249	3
Other	$5,910,200	14
Total	$3,782,380,113	122

1) SELFISH MINING ATTACKS

Selfish mining is a data consistency attack [17] where the attacker tries to steal the mining reward. The default behaviour of nodes in the blockchain is to announce a block immediately once it is found, instead of the attacker (e.g., selfish miner or mining pool) keeps the block in his private chain [54]. The attacker continues to extend his private chain until one step ahead of the main chain. If the attacker gets lucky and finds the next block before honest miners, then the longest chain rule will make the attacker chain longer, and the nodes will switch to the new longest chain. The attacker will get a reward for his mined blocks, but the blocks generated by the other miners become orphans and confer no reward.

To overcome selfish mining attacks, the authors [129] use the PoR-based consensus mechanism. Nicolas *et al.* [17] conduct a literature review and gather the countermeasures for mitigating selfish mining attacks. The countermeasures are categorised under six defensive strategies. The study [130] utilised the notion of "truth state" for blocks by adding the expected confirmation height parameter in each transaction data structure to recognise selfish mining in the network.

2) QUANTUM COMPUTING THREATS

Blockchain systems are vulnerable to quantum computing that could solve elliptic curve digital algorithm (ECDSA) [55]. Mostly, blockchain systems (e.g., Bitcoin) use ECDSA for transaction authentication. Quantum computing can threaten blockchain systems in two different aspects. First, the inversion of hashes compromises the authenticity of transactions in the blockchain. Second, break the encryption security, such as public/private key cryptography, digital signatures, or encrypted communication.

The researchers are trying to eliminate threats posed by quantum computing. Some quantum computing resistant cryptography schemes already exist (e.g., lattice-based, multivariate, hash-based, code-based cryptography, and new schemes are appearing) [131]. Yin *et al.* [56] implemented the anti-quantum transaction authentication scheme by incorporating lattice-based cryptography. The authors [55] present the post-quantum blockchain (using lattice-based delegation algorithm) and cryptocurrency scheme on it.

3) SMART CONTRACT ATTACKS

Smart contract attacks are increasing in decentralised applications [57] and usually categorised under application bugs when developers fail to identify code errors. The attacker exploits these coding errors and damages the blockchain systems. In 2016, the DAO emerged [58] as an open-source SC on Ethereum platform (now Ethereum classic) aiming to automate transactional processes by eliminating human input. The participants could exchange DAO tokens with Ether and cast their vote on propositions to earn rewards. The DAO SC contained severe coding vulnerabilities, including recursive (reentrancy) calls (Fig. 18). The attacker exploits the reentrancy vulnerability and gained control on $60 million Ethers [3], [4].

To prevent SCs attacks, apply defensive programming techniques and use testing tools to detect vulnerabilities in SCs. For example, the authors [127] present the penetration testing framework for SCs to use in order to test dApps before deployment. According to the findings, the proposed

```
function revoke() remote{
   uint256 value = balances[msg.sender];
   require(msg.sender.call.value(value)());
   balances[msg.sender] = 0;
}
```

FIGURE 18. Function with a reentrancy vulnerability [57].

penetration testing system discovered vulnerabilities that were missed by traditional automated penetration testing scanners. The study [57] presents a few tools to detect SCs vulnerabilities that the attacker can exploit. Furthermore, the automated tool [3] that uses run-time trace analysis to identify smart contract bugs. Tikhomirov *et al.* [128] developed a static analysis method for detecting reentrancy bugs in smart contracts. The method converts solidity source code into an XML-based intermediate representation and validates it using XPath patterns.

4) USER WALLET ATTACKS

Blockchain user wallets are suspects to various security threats where the attacker targets the user wallet credentials. In 2017, the attacker (alias norbertvdberg) created an online 81-digit IOTA seed generator (iotaseed.io) and conducted a phishing campaign with this service. Many users use this fake seed generator to create an IOTA seed that allows accessing their IOTA wallet. On 19 January 2018, the attacker started sending IOTA coins from the compromised seeds to his wallet. As a result, the attacker gains access to more than $11.4 million worth of IOTA coins [59]. In [60] discussed a dictionary attack to break cryptographic hash and salt by trying a hash of common passwords (e.g., password1) to find user wallet credentials.

Moreover, the attacker can exploit weak or vulnerable hash functions and digital signatures. For example, IOTA insecure Curl hash function [61] and low entropy ECDSA [62]. The group of researchers exploited Nano S Ledger wallet vulnerability [63]. The researchers use the malware to replace the funds receive address with their address. The vulnerability exists because Nano S Ledger had no viable option to verify the receiving address's integrity. As a result, the researchers obtained users private keys, PINs, recovery seeds, and passphrases from the wallet. Moreover, the unauthorised access to native XRP wallets and theft of crypto-assets [64]. In 2019, the attacker stole 342,000 ETH coins from UPbit cold wallet [64] and over $4.5 million in XRP and $237,500 in ADA from Bitrue hot wallets [64].

To protect against attacks on user wallets, the authors [132] suggest using multi-signature wallets that require two or more private keys to sign and send a transaction and secure cloud storage systems to store private keys. Use hardware security modules (HSMs) to store and manage private, and public keys [133]. The HSMs use a security-focused operating system and prevent unauthorised contact to the user wallet. Also, use a cold wallet (e.g., paper wallet), and when using an online service to generate a wallet, verify the authenticity and legitimacy of the service.

5) ENDPOINT VULNERABILITIES

Blockchain systems include digital wallets, hardware wallets, private keys, passwords, client-side applications. Therefore, the protection of these endpoints is utmost, and if any of these endpoints is compromised, the attacker gains access to the user digital assets [65]. Nevertheless, endpoint vulnerabilities remain susceptible through social engineering, phishing, real-world theft, or physical access to user wallet, phone or computer.

To minimise endpoint vulnerabilities, use multi-signature wallets, cold wallets, and do not share private keys of wallets with anyone [65]. The users can educate themselves to learn about social engineering approaches, use authentic and legit sources to protect against phishing, also HSMs can protect against endpoint vulnerabilities [65].

6) PLATFORM SPECIFIC ATTACKS

Various blockchain platforms have (known or hidden) vulnerabilities that attacker can exploit. For example, smart contracts code bugs still exist on the Ethereum platform because of the immutability principle and cannot be changed. The attacker can discover these code-related vulnerabilities and steal digital assets [57]. Furthermore, suppose Ethereum smart contract is missing (or incorrectly used) a modifier on a function. In that case, it lets the attacker become a contract owner that enables him to use sensitive functionality in the contract [66].

To overcome platform-specific attacks, follow the best practices and guidelines to implement the dApp. Use defensive programming techniques and testing tools [127] to detect vulnerabilities in SCs before deployment. Also, write upgradable SCs that allow updating SCs after deployment [134] to fix known coding bugs that the attacker could exploit.

B. IMPLEMENTATION CHALLENGES

Blockchain emerges to overcome traditional applications' security challenges along with excluding third-party intervention in a transaction process. In reality, there requires extensive work to be done before the blockchain can fulfil its promises. Along with security challenges, blockchain faces various implementation challenges, e.g., scalability, interoperability, regulatory issues, and scams.

1) SCALABILITY

Blockchain transaction rate (throughput) is slow. For example, the Bitcoin process only 7 TPS, Ethereum 15 TPS, and permissioned blockchain HLF can process 3500+ TPS. In contrast, VISA performs 1700-2000 TPS (capable for 65000+ TPS) [82]. If blockchain is to be a disruptive technology, then along with the inherent blockchain-related advantages, it should be robust at high speeds.

Various solutions are already available to overcome scalability issues. For example, the study [70] presents the proposals (e.g., lighting protocol, sharding, super quadratic sharding, DPoS) to solve the scalability issues in the blockchain. The authors [71] analyse various techniques (e.g., On-chain, off-chain, side-chain, child-chain, inter-chain) to enhance scalability. However, the authors conclude that more work is required in this field.

2) INTEROPERABILITY

Blockchain interoperability generally tackles the ability to share states and transacting across different chains [69]. Currently, there is no sophisticated (or no direct) method to allow one blockchain to transmit information to another blockchain [68], [69].

The researchers are working to provide a sustainable ecosystem for interoperability [68]. The study presents an interoperable blockchain architecture that standardised common components of blockchain to achieve a higher degree of interoperability. The XCLAIM (cross-claim) framework is presented [69] to achieve a trustless and efficient cross-chain assets exchange.

3) REGULATORY ISSUES

The decentralisation brings benefits but also poses legal and regulatory challenges. For example, there is no centralised party in permissionless blockchain systems that takes responsibility for the provision of services, controls, and associated data sets. Moreover, no standardised jurisdiction and laws to regulate blockchain-enabled transactions. Each country has its own jurisdiction that overwhelms the number of laws and regulations for blockchain-enabled transactions [72].

According to [73], permissionless blockchain systems are incompatible with privacy laws (e.g., EU general data protection regulation (GDPR)). There are various aspects that conflict with GDPR. For example, no one is held accountable for a blockchain's availability or security, and anybody can access the data on the network. Under GDPR users are controllers to their data but the immutability property of blockchain cannot let the user delete (or update) their data. In governance, a key question for regulators is who should be held accountable for breaches of laws and regulations. Moreover, mostly permissionless blockchain systems use crypto coins as a payment mode to pay for goods or services. Permissionless blockchain systems do not have a controlling authority, and the transaction is pseudo-anonymous. Hence, no one (or difficult) to hold accountable for the taxation, and it raised concerns for tax authorities.

Currently, there is uncertainty about regulatory requirements related to blockchain applications. However, to minimise the regulatory issues, various organisations are working together to define the regulatory guidance. For instance, a standardised legal framework for blockchains regarding data stored in blockchains, the legal validity of financial instruments issued in blockchains, legal nature of blockchains and shared distributed ledgers [135].

4) SCAMS

Initial coins offering (ICO) is a crowdfunding method to raise funding for a project. In 2017, Bloomberg research [74] identified that 78% of ICOs were scams or fraudulent, 4% failed, 3% had gone dead, and only 15% listed on exchanges for active trading. In 2018, 1,000 ICO projects failed, losing $100 million of investors [75].

To mine cryptocurrencies, several companies started providing cloud mining services to rent their server/hardware. In 2019, the attacker stole $722 million using BitClub Network [76]. Fake exchanges [77] that give big bonuses in start to deposit crypto assets. Once a user deposits, then such exchanges either steal, charge a high fee for withdrawing or make hard to withdraw. Ponzi or pyramid schemes promise high returns to investors but run away when getting sufficient profits [78]. For example, Plus Token scam when the attacker withdraws over 3 Billion dollars in cryptocurrencies (Bitcoin, Ethereum, and EOS) and leaves the message "sorry we have run." [79]. Silk Road dark web [80] that operated over Tor hidden service used bitcoin as payment. Similarly, the ransomware attack that encrypts files and shows pop up to send a certain amount of bitcoins in exchange for decrypt key [81].

The possible ways to avoid scams are to be aware of fake exchanges and wallets and verify authenticity and legitimacy before investing in ICO projects or cryptocurrency. The users can educate themselves to learn about social engineering approaches, protect against phishing, use multi-signature wallets, cold wallets, and implement HSMs [65]. Also, the regulatory guidance and standardised legal framework for blockchains will help to limit blockchain-based scams.

The list of such security and implementation challenges of blockchain systems is long, and new challenges are emerging that hurdle the acceptance of blockchain technology. Permissioned blockchain systems are emerging to overcome a few of these challenges.

VI. PERMISSIONED BLOCKCHAIN SYSTEMS

Permissioned blockchain systems include only pre-verified nodes in the network, and the access control layer governs the actions of participants. These systems are energy-efficient because they do not require energy-waste consensus. Provide a high rate of transactions per second (TPS), decentralised operations, and data storage in the strong governance structure. Also, ensure user privacy, the confidentiality of transactions and cost-effective transactions by removing mining fees. Permissioned blockchain systems are emerging in industry-level enterprises and businesses where security and privacy are imperative. Also, overcome various security and implementation challenges of permissionless blockchain systems. The prominent permissioned blockchain systems are HLF, Corda, and EoS.

A. SYBIL AND DOUBLE-SPENDING

In a permissioned blockchain, the nodes are required to prove their identity before joining the network. The network includes a limited number of nodes, so participants are known to each, and an access control layer controls participants operations [13]. For example, HLF verifies nodes before joining the network and creates a list of whitelisted nodes [83]. The node that only belongs to whitelisted nodes can participate in the consensus process, and a built-in access control layer controls their read/write operations. All such premises of permissioned blockchain systems help to eliminate the risks of Sybil attack [83], [84].

Permissioned blockchain systems utilise pluggable consensus and do not indulge in a mining process. Nodes are pre-verified that eliminate the risk of Sybil identities, nodes cannot fork ledger or maintain private fork, and contains transaction monitoring tool. For example, HLF supports pluggable (e.g., PBFT or CFT) consensus [86], [87] that enables the dApp to be more effectively customised to fit particular use cases and trust models according to the business needs. Moreover, in HLF, all nodes maintain the single ledger, follows an execute-order-validate mechanism [85] where transaction flow is divided into three steps: i) executing a transaction and checking its correctness ii) ordering transactions through a consensus protocol, and iii) validation of transaction(s). Also, HLF dApps can incorporate a transaction manager [85], an automated transaction monitoring tool. It performs transaction validation to catch conflicting transactions (or conflicting read-write checks). The transaction manager maintains the state in a versioned key-value store and validates all the transactions sequentially.

B. OTHER SECURITY CHALLENGES

Permissioned blockchain systems do not require mining-based consensus that eliminates selfish mining and other mining-related risks. Permissioned blockchain systems implement HSMs, and participants are known to each other, thus eliminating the risk of user wallet attacks. However, permissioned blockchain systems are vulnerable to quantum computing threats, smart contract attacks, endpoint vulnerabilities [65], and platform-specific [67] attacks. For example, the blockchain-based Corda platform is vulnerable to denial of state attack [67] where a node knowingly builds an invalid transaction consuming some set of existing states. The countermeasures that are discussed above (in Section V-A) can restrict these attacks in permissioned blockchains.

C. IMPLEMENTATION CHALLENGES

Permissioned blockchains systems provide high transaction throughput and are moderately scalable. Compared to Bitcoin and Ethereum, HLF is capable of performing 3500+ TPS and Corda 1678 TPS (Table 2). Permissioned blockchain systems perform decentralised operations and data storage in governance structure, comprise clear ownership and responsibility roles. Thus, eliminating the regulatory issues and risks of various scams. However, the interoperability issue exists in permissioned blockchain systems where the information access across various blockchain systems is unattainable.

The security and implementation challenges of permissioned blockchains compared to permissionless blockchain systems are relatively low or controlled to a certain level. The attacks are less beneficial and burdensome. For example, permissioned blockchain systems do not directly include monetary assets, pre-verification before joining the network, and control the participant's operations by permissioned settings and access control layer. These concepts lead to permissioned blockchain adoption among industry-level enterprises (e.g., logistic partners, supply vendors, financial institutions, etc). Because security, user privacy, and transaction confidentiality are important aspects along with decentralisation.

VII. DISCUSSION

Security risk management is an iterative process because new security risks, threats and vulnerabilities could emerge. For example, BitMex research reported the double-spent transaction in Bitcoin on the 21st of Jan 2021 [88]. The source and attack method of this double-spend transaction is not yet known. However, it indicates that the attacker continuously builds new techniques to sabotage the integrity, confidentiality and availability of the blockchain systems. To communicate security risks to blockchain developers, practitioners, and other associated stakeholders, we need a comprehensive blockchain security reference model.

A. FUTURE WORK

In our *future work*, we aim to build an ontology-based blockchain security reference model as a security risk management tool to evaluate the security needs of blockchain systems systematically. In our previous work [96], as a proof of concept, we build a Corda-based security ontology *(CordaSecOnt)* to improve the security of the financial industry from an ontological analysis that combines blockchain-based Corda platform. The CordaSecOnt uses the Web ontology language (OWL) to build a semantic knowledge base to eliminate conceptual ambiguity and a semantic gap in the information security of the financial industry. The CordaSecOnt utilises the SRM domain model and provides the classifications of assets, security criteria, threats, vulnerabilities, risk treatments, security requirements, countermeasures.

Ontology-based security reference model activates the dynamic and seamless process to add new knowledge in the blockchain systems security domain. Similar to this research and CordaSecOnt, the ontology-based blockchain security reference model will also utilise the SRM domain model to explore the assets to secure, security risks, threats, vulnerabilities, and countermeasures.

Similar to the ontology-based security reference model, it is possible to implement this framework as a decision support tool. The tool would help to decide on the security countermeasures based on the security threats and their vulnerabilities. The process to construct a decision support tool will follow the guidelines of the SRM domain model. It also enables a dynamic and seamless approach to add new knowledge and interpreting the security risks of blockchain systems. For example, to describe the assets to secure, security risks, threats, vulnerabilities, and countermeasures of blockchain-based applications.

B. THREATS TO VALIDITY

To ensure the quality of empirical studies in software engineering, assessing threats to validity is important [95]. Our current research has several limitations, and we discussed following the Zhou *et al.* [95] threats to validity mapping. Threats to validity that are relevant to this research are restricted time span, publication bias, subjective interpretation, and lack of expert evaluation.

The *restricted time-span* is that the researcher cannot predict other applicable studies beyond the time span. For example, blockchain is relatively new but continuously evolving, and not all the possible security risks are researched. It reflects the likelihood that a wide variety of security risks will emerge in the future. The threat concerning *publication bias* is that the related studies are more likely to report positive results than negative results. Also, various security risks and their countermeasures are unclear, or there is no real implementation available. The threat of *subjective interpretation* exists since we might have different interpretations and opinions related to identified threats, vulnerabilities, and countermeasures of defined security risks. Moreover, a *lack of expert evaluation* may also lead to a subjective interpretation and erroneous conclusion.

Blockchain technology looks promising but still in its infancy. Many blockchain systems have recently appeared, but their security risks need to be evaluated on a larger scale. We are developing an ontology-based blockchain security reference model, and it may solve the threats to validity. Ontology by design is dynamic, and researchers can update at any time, resolving the restricted time span and publication bias threats. Furthermore, the practitioners and researchers will be able to study and update the ontology online. As a result, the threats related to subjective interpretation and lack of expert evaluation will be minimised. Overall, resolving the above-mentioned threats to validity could bring richer insights and lead to a more in-depth contribution to performing SRM of blockchain systems.

VIII. CONCLUDING REMARKS

Sybil and Double-Spending risks are emerging in blockchain systems, and security risk management enables a mechanism to explore these risks and enforce their countermeasures. This paper presents a framework based on the security risk management domain model for exploring Sybil and Double-spending risks in blockchain systems. The framework illustrates the protected assets or assets to be protected, the classification of threats that the attacker could trigger using Sybil attack, the identification of threats that cause Double-spending, the vulnerabilities of identified threats, and their countermeasures. We evaluated a newly built framework by exploring Sybil and Double-spending risks in Ethereum-based healthcare dApps. Also, we provide emerging security and implementation challenges of blockchain systems and the role of permissioned blockchain systems. To the end, we present the future research directions and threats to validity. This research's findings could support software developers and decision-makers regarding Sybil and Double-spending risks while building blockchain systems.

ACKNOWLEDGMENT

The European Commission support (under grant no. 621646-EPP-1-2020-1-FR-EPPKA2-SSA-B) for the production of this publication does not constitute endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

REFERENCES

[1] T. Sato and Y. Himura, "Smart-contract based system operations for permissioned blockchain," in *Proc. 9th IFIP Int. Conf. New Technol., Mobility Secur. (NTMS)*, Feb. 2018, pp. 1–6.
[2] M. Crosby, Nachiappan, P. Pattanayak, S. Verma, and V. Kalyanaraman, "BlockChain technology beyond bitcoin," Dept. Eng., Sutardja Center Entrepreneurship Technol., Berkeley, CA, USA, 2015, pp. 1–35.
[3] C. Liu, H. Liu, Z. Cao, Z. Chen, B. Chen, and B. Roscoe, "ReGuard: Finding reentrancy bugs in smart contracts," in *Proc. 40th Int. Conf. Softw. Eng., Companion*, May 2018, pp. 65–68.
[4] N. Atzei, M. Bartoletti, and T. Cimoli, "A survey of attacks on Ethereum smart contracts (SoK)," in *Proc. Int. Conf. Princ. Secur. Trust*, 2017, pp. 1–24.
[5] M. Iqbal and R. Matulevicius, "Comparison of blockchain-based solutions to mitigate data tampering security risk," in *Business Process Management: Blockchain and Central and Eastern Europe Forum*. Cham, Switzerland: Springer, 2019, pp. 13–28.
[6] M. Iqbal and R. Matulevicius, "Blockchain-based application security risks: A systematic literature review," in *Proc. Adv. Inf. Syst. Eng. Workshops*, 2019, pp. 176–188.
[7] A. Mohaisen, "The Sybil attacks and defenses: A survey," *Smart Comput. Rev.*, vol. 3, no. 6, pp. 1–10, Dec. 2013.
[8] M. Nesbitt. (2019). *Deep Chain Reorganization Detected on Ethereum Classic (ETC)*. https://blog.coinbase.com/ethereum-classic-etc-is-currently-being-51-attacked-33be13ce32de
[9] É. Dubois, N. Mayer, P. Heymans, and R. Matulevièius, "A systematic approach to define the domain of information system security risk management," in *Intentional Perspectives on Information Systems Engineering*. Berlin, Germany: Springer, 2010, pp. 289–306.
[10] R. Matulevicius, *Fundamentals of Secure System Modelling*. New York, NY, USA: Springer, 2017.
[11] M. Brunner, C. Sauerwein, M. Felderer, and R. Breu, "Risk management practices in information security: Exploring the status quo in the DACH region," *Comput. Secur.*, vol. 92, May 2020, pp. 1–18.
[12] Y. Yu, V. N. L. Franqueira, T. T. Tun, R. J. Wieringa, and B. Nuseibeh, "Automated analysis of security requirements through risk-based argumentation," *J. Syst. Softw.*, vol. 106, pp. 102–116, Aug. 2015.
[13] S. Ali, G. Wang, B. White, and R. L. Cottrell, "A blockchain-based decentralized data storage and access framework for PingER," in *Proc. 17th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun., 12th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, Aug. 2018, pp. 1303–1308.
[14] V. Buterin. (2014). *A Next-Generation Smart Contract and Decentralized Application Platform*. [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper
[15] D. Macrinici, C. Cartofeanu, and S. Gao, "Smart contract applications within blockchain technology: A systematic mapping study," *Telematics Inform.*, vol. 35, no. 8, pp. 2337–2354, 2018.
[16] L. W. Cong, Z. He, and J. Zheng, "Blockchain disruption and smart contracts," *SSRN Electron. J.*, vol. 32, no. 5, pp. 1–54, 2017.
[17] K. Nicolas, S. Member, Y. Wang, G. C. Giakos, and B. Wei, "Blockchain system defensive overview for double-spend and selfish mining attacks: A systematic approach," *IEEE Access*, vol. 9, pp. 3838–3857, 2020.
[18] J. R. Douceur, "The Sybil attack," in *Proc. Int. Workshop Peer Peer Syst.*, 2002, pp. 251–260.
[19] J. K. So and D. S. Reeves, "Defending against Sybil nodes in BitTorrent," in *Proc. Int. Conf. Res. Netw.*, 2011, pp. 25–39.
[20] M. Saad, J. Spaulding, L. Njilla, C. Kamhoua, S. Shetty, D. Nyang, and D. Mohaisen, "Exploring the attack surface of blockchain: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1977–2008, 3rd Quart., 2020.
[21] J. H. Mosakheil. (2018). *Security Threats Classification in Blockchains*. [Online]. Available: https://repository.stcloudstate.edu/msia_etds/48
[22] C. Pinzón and C. Rocha, "Double-spend attack models with time advantange for bitcoin," *Electron. Notes Theor. Comput. Sci.*, vol. 329, pp. 79–103, Dec. 2016.
[23] S. Sayeed and H. Marco-Gisbert, "Assessing blockchain consensus and security mechanisms against the 51% attack," *Appl. Sci.*, vol. 9, no. 9, p. 1788, Apr. 2019.
[24] S. Zhang and J.-H. Lee, "Double-spending with a Sybil attack in the bitcoin decentralized network," *IEEE Trans. Ind. Informat.*, vol. 15, no. 10, pp. 5715–5722, Oct. 2019.
[25] T. Rajab, M. Hossein Manshaei, M. Dakhilalian, M. Jadliwala, and M. A. Rahman, "On the feasibility of Sybil attacks in shard-based permissionless blockchains," 2020, *arXiv:2002.06531*. [Online]. Available: http://arxiv.org/abs/2002.06531
[26] P. Otte, M. de Vos, and J. Pouwelse, "TrustChain: A Sybil-resistant scalable blockchain," *Future Gener. Comput. Syst.*, vol. 107, pp. 770–780, Jun. 2020.
[27] S. Pradhan, S. Tripathy, and S. Nandi, "Blockchain based security framework for P2P filesharing system," in *Proc. IEEE Int. Conf. Adv. Netw. Telecommun. Syst. (ANTS)*, Dec. 2018, pp. 1–6.
[28] M. Quintyne-Collins. (2019). *Short Paper: Towards Characterizing Sybil Attacks in Cryptocurrency Mixers*. [Online]. Available: https://eprint.iacr.org/2019/1111.pdf
[29] P. Wei, Q. Yuan, and Y. Zheng, "Security of the blockchain against long delay attack," in *Advances in Cryptology—ASIACRYPT*. Brisbane, QLD, Australia: Springer, 2018, pp. 250–275.
[30] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on Bitcoin's peer-to-peer network," in *Proc. 24th USENIX Secur. Symp.*, 2015, pp. 129–144.
[31] C. Pérez-Solà, S. Delgado-Segura, G. Navarro-Arribas, and J. Herrera-Joancomartí, "Double-spending prevention for bitcoin zero-confirmation transactions," *Int. J. Inf. Secur.*, vol. 18, no. 4, pp. 451–463, Aug. 2019, doi: 10.1007/s10207-018-0422-4.
[32] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "MedRec: Using blockchain for medical data access and permission management," in *Proc. 2nd Int. Conf. Open Big Data (OBD)*, Aug. 2016, pp. 25–30.
[33] L. Zhou, L. Wang, and Y. Sun, "MIStore: A blockchain-based medical insurance storage system," *J. Med. Syst.*, vol. 42, no. 8, pp. 148–165, Aug. 2018.
[34] K. Jonathan and A. K. Sari, "Security issues and vulnerabilities on a blockchain system: A review," in *Proc. Int. Seminar Res. Inf. Technol. Intell. Syst. (ISRITI)*, Dec. 2019, pp. 228–232.
[35] K. Wust and A. Gervais, "Do you need a blockchain?" in *Proc. Crypto Valley Conf. Blockchain Technol. (CVCBT)*, Jun. 2018, pp. 45–54.
[36] S. Farshidi, S. Jansen, S. España, and J. Verkleij, "Decision support for blockchain platform selection: Three industry case studies," *IEEE Trans. Eng. Manag.*, vol. 67, no. 4, pp. 1109–1128, Nov. 2020.
[37] E. Attah. (2019). *Five Most Prolific 51% Attacks in Crypto: Verge, Ethereum Classic, Bitcoin Gold, Feathercoin, Vertcoin*. [Online]. Available: https://cryptoslate.com/prolific-51-attacks-crypto-verge-ethereum-classic-bitcoin-gold-feathercoin-vertcoin
[38] E. Deirmentzoglou, G. Papakyriakopoulos, and C. Patsakis, "A survey on long-range attacks for proof of stake protocols," *IEEE Access*, vol. 7, pp. 28712–28725, 2019.
[39] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 17–30.
[40] *What Is a Dusting Attack?* [Online]. Available: https://academy.binance.com/en/articles/what-is-a-dusting-attack
[41] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and Cryptocurrency Technologies*. Princeton, NJ, USA: Princeton Univ. Press, 2016.
[42] N. V. Saberhagen. (2013). *CryptoNote V 2.0*. [Online]. Available: https://bytecoin.org/old/whitepaper.pdf
[43] M. Rosenfeld, "Analysis of hashrate-based double spending," 2014, *arXiv:1402.2009*. [Online]. Available: http://arxiv.org/abs/1402.2009
[44] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. [Online]. Available: https://bitcoin.org/bitcoin.pdf
[45] M. Rosenfeld. (2012). *What is a Finney Attack?* [Online]. Available: https://bitcoin.stackexchange.com/questions/4942/what-is-a-finney-attack
[46] Y. Marcus, E. Heilman, and S. Goldberg, "Low-resource eclipse attacks on ethereum's peer-to-peer network," IACR ePrint Cryptol. Rep., Boston, MA, USA, Tech. Rep. 236, 2018, vol. 2018.
[47] M. Apostolaki, A. Zohar, and L. Vanbever, "Hijacking bitcoin: Routing attacks on cryptocurrencies," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 375–392.
[48] G. O. Karame, E. Androulaki, and S. Capkun, "Double-spending fast payments in bitcoin," in *Proc. ACM Conf. Comput. Commun. Secur. (CCS)*, 2012, pp. 906–917.
[49] Coinbase. *Bitcoin Glossary*. Accessed: Feb. 17, 2021. [Online]. Available: https://help.coinbase.com/en/coinbase/getting-started/crypto-education/bitcoin-glossary

[50] T. Bamert, C. Decker, L. Elsen, R. Wattenhofer, and S. Welten, "Have a snack, pay with bitcoins," in *Proc. IEEE P P*, Sep. 2013, pp. 1–5.

[51] Q. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du, and M. Guizani, "MeDShare: Trust-less medical data sharing among cloud service providers via blockchain," *IEEE Access*, vol. 5, pp.14757–14767, 2017.

[52] P. Zhang, J. White, D. C. Schmidt, G. Lenz, and S. T. Rosenbloom, "FHIRChain: Applying blockchain to securely and scalably share clinical data," *Comput. Structural Biotechnol. J.*, vol. 16, pp. 267–278, 2018.

[53] C. Ruth. (2021). *Blockchain Hackers Stole $3.8 Billion in 122 Attacks Throughout 2020*. [Online]. Available: https://atlasvpn.com/blog/blockchain-hackers-stole-3-8-billion-in-122-attacks-throughout-2020

[54] S. Bag, S. Ruj, and K. Sakurai, "Bitcoin block withholding attack: Analysis and mitigation," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 8, pp. 1967–1978, Aug. 2017.

[55] Y.-L. Gao, X.-B. Chen, Y.-L. Chen, Y. Sun, X.-X. Niu, and Y.-X. Yang, "A secure cryptocurrency scheme based on post-quantum blockchain," *IEEE Access*, vol. 6, pp. 27205–27213, 2018.

[56] W. Yin, Q. Wen, W. Li, H. Zhang, and Z. Jin, "An anti-quantum transaction authentication approach in blockchain," *IEEE Access*, vol. 6, pp. 5393–5401, 2018.

[57] S. Sayeed, H. Marco-Gisbert, and T. Caira, "Smart contract: Attacks and protections," *IEEE Access*, vol. 8, pp. 24416–24427, 2020.

[58] D. Peters, J. Wetzlich, F. Thiel, and J.-P. Seifert, "Blockchain applications for legal metrology," in *Proc. IEEE Int. Instrum. Meas. Technol. Conf. (I MTC)*, May 2018, pp. 1–6.

[59] A. Studer and Iotaseed.io. (2018). *The History of IoT Aseed.io—A Malicious IOTA Seed Generator*. [Online]. Available: https://iotaseed.io/iota-seed-generator-scam

[60] P. Praitheeshan, Y. W. Xin, L. Pan, and R. Doss, "Attainable hacks on Keystore files in Ethereum wallets—A systematic analysis," in *Proc. Int. Conf. Future Netw. Syst. Secur.*, 2019, pp. 99–117.

[61] N. Narula. (2017). *Cryptographic Vulnerabilities in IOTA*. [Online]. Available: https://medium.com/@neha/cryptographic-vulnerabilities-in-iota-9a6a9ddc4367

[62] Z. Wang, H. Yu, Z. Zhang, J. Piao, and J. Liu, "ECDSA weak randomness in bitcoin," *Future Gener. Comput. Syst.*, vol. 102, pp. 507–513, Jan. 2020.

[63] A. Zmudzinski. (2018). *Research Team Demonstrates Hard Wallets Vulnerabilities*. [Online]. Available: https://cointelegraph.com/news/research-team-demonstrates-hard-wallets-vulnerabilities-trezor-promises-firmware-update

[64] *A Complete List of Cryptocurrency Exchange Hacks [Updated]*. Accessed: Jan. 5, 2021. [Online]. Available: https://blog.idex.io/all-posts/a-complete-list-of-cryptocurrency-exchange-hacks-updated

[65] J. Velissarios, J. Herzig, and D. Unal, "Blockchain's potential starts with security," in *Accenture: Blockchain's Potential Starts With Security*. San Francisco Bay Area, CA, USA: Accenture, 2019.

[66] *Unprotected Function*. Accessed: Feb. 23, 2021. [Online]. Available: https://github.com/crytic/not-so-smart-contracts/tree/master/unprotected_function

[67] T. Koens, S. King, M. V. D. Bos, C. V. Wijk, and A. Koren, "Solutions for the corda security and privacy trade-off: Having your cake and eating it," ING, Amsterdam, The Netherlands, Internal Rep. 3024436, 2019.

[68] T. Hardjono, A. Lipton, and A. Pentland, "Toward an interoperability architecture for blockchain autonomous systems," *IEEE Trans. Eng. Manag.*, vol. 67, no. 4, pp. 1298–1309, Nov. 2020.

[69] A. Zamyatin, D. Harz, J. Lind, P. Panayiotou, A. Gervais, and W. Knottenbelt, "XCLAIM: Trustless, interoperable, cryptocurrency-backed assets," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 193–210.

[70] A. Chauhan, O. P. Malviya, M. Verma, and T. S. Mor, "Blockchain and scalability," in *Proc. IEEE Int. Conf. Softw. Qual., Rel. Secur. Companion (QRS-C)*, Jul. 2018, pp. 122–128.

[71] S. Kim, Y. Kwon, and S. Cho, "A survey of scalability solutions on blockchain," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2018, pp. 1204–1207.

[72] P. Yeoh, "Regulatory issues in blockchain technology," *J. Financial Regulation Compliance*, vol. 25, no. 2, pp. 196–208, May 2017.

[73] J. Salmon and G. Myers, "Blockchain and associated legal issues for emerging markets," in *Fresh Ideas About Business in Emerging Markets*. London, U.K.: IFC World Bank Group, 2019.

[74] S. Dowlat and Bloomberg. (2018). *Cryptoasset Market Coverage Initiation: Network Creation*. [Online]. Available: https://bit.ly/3rXgwlD

[75] O. O. Emmanuel. (2018). *ICO Exit Scams: Bad Actors Cart Away $100 Million in 2018*. [Online]. Available: https://btcmanager.com/ico-exit-scams-bad-actors-cart-away-100-million-in-2018

[76] U. S. Attorney's Office. (2019). *Three Men Arrested in $722 Million Cryptocurrency Fraud Scheme*. [Online]. Available: http://bit.ly/3qrVD1R

[77] F. Erazo. (2020). *The Leaders of a Fake Crypto Exchange Are Now Behind Bars*. [Online]. Available: https://cointelegraph.com/news/the-leaders-of-a-fake-crypto-exchange-are-now-behind-bars

[78] SEC Office of Investor Education and Advocacy. *Ponzi Schemes Using Virtual Currencies*. Accessed: Feb. 23, 2021. [Online]. Available: https://www.sec.gov/investor/alerts/ia_virtualcurrencies.pdf

[79] Michael. (2020). *Plus Token (PLUS) Scam—Anatomy of a Ponzi*. [Online]. Available: https://boxmining.com/plus-token-ponzi/

[80] Wikipedia. *Silk Road (Marketplace)*. Accessed: Nov. 15, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Silk_Road_(marketplace)

[81] R. Whitwam. (2017). *NotPetya Ransomware Hackers Want 100 Bitcoins for Decryption Keys*. [Online]. Available: http://bit.ly/3rV8yJZ

[82] Visa.co.uk. *Visa Fact Sheet*. Accessed: Feb. 26, 2021. [Online]. Available: https://bit.ly/3bfktvz

[83] P. Gallo, S. Pongnumkul, and U. Q. Nguyen, "BlockSee: Blockchain for IoT video surveillance in smart cities," in *Proc. IEEE Int. Conf. Environ. Electr. Eng., IEEE Ind. Commercial Power Syst. Eur. (EEEIC/I CPS Eur.)*, Jun. 2018, pp. 1–6.

[84] R. Nygaard, H. Meling, and L. Jehl, "Distributed storage system based on permissioned blockchain," in *Proc. 34th ACM/SIGAPP Symp. Appl. Comput.*, Apr. 2019, pp. 338–340.

[85] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, and S. Muralidharan, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, Apr. 2018, pp. 1–15.

[86] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan, "BLOCKBENCH: A framework for analyzing private blockchains," in *Proc. ACM Int. Conf. Manage. Data*, May 2017, pp. 1085–1100.

[87] Hyperledger. (2019). *Hyperledger Fabric Introduction*. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-1.4/whatis.html

[88] BitMEX Fork Monitor. (2021). *Stale Block Candidates at Height 666833*. [Online]. Available: https://forkmonitor.info/stale/btc/666833

[89] Honeyminer.com. (2019). *What's a Blockchain Reorg?*. [Online]. Available: https://blog.honeyminer.com/whats-a-blockchain-reorg

[90] Y. Du, J. Liu, Z. Guan, and H. Feng, "A medical information service platform based on distributed cloud and blockchain," in *Proc. IEEE Int. Conf. Smart Cloud (SmartCloud)*, Sep. 2018, pp. 34–39.

[91] R. Pass and E. Shi, "FruitChains: A fair blockchain," in *Proc. ACM Symp. Princ. Distrib. Comput.*, Jul. 2017, pp. 315–324.

[92] J. Ahn, "EdenChain: The programmable economy platform," Eden, Singapore, White Paper V 1.2, 2018.

[93] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," Keele Univ. Durham Univ., Keele, U.K., Tech. Rep. EBSE 2007-001, 2007.

[94] P. Swathi, C. Modi, and D. Patel, "Preventing Sybil attack in blockchain using distributed behavior monitoring of miners," in *Proc. 10th Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT)*, Jul. 2019, pp. 6–11.

[95] X. Zhou, Y. Jin, H. Zhang, S. Li, and X. Huang, "A map of threats to validity of systematic literature reviews in software engineering," in *Proc. 23rd Asia–Pacific Softw. Eng. Conf. (APSEC)*, 2016, pp. 153–160.

[96] M. Iqbal and R. Matulevičius, "Corda security ontology: Example of post-trade matching and confirmation," *Baltic J. Modern Comput.*, vol. 8, no. 4, pp. 638–674, 2020.

[97] M. Tran, I. Choi, G. J. Moon, A. V. Vu, and M. S. Kang, "A stealthier partitioning attack against bitcoin peer-to-peer network," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2020, pp. 894–909.

[98] E. Zaghloul. (2018). *Beginners Guide on Blockchain Security Attacks Part 1—Network*. [Online]. Available: https://medium.com/zkcapital/beginners-guide-on-blockchain-security-attacks-part-1-network-ca4e74435723

[99] T. Baumeister, Y. Dong, Z. Duan, and G. Tian, "A routing table insertion (RTI) attack on freenet," in *Proc. Int. Conf. Cyber Secur.*, Alexandria, Egypt, Dec. 2012, pp. 8–15.

[100] L. Coleman. (2016). *Ethereum Responds to Recent DDoS Attack*. [Online]. Available: https://www.ccn.com/ethereum-responds-to-recent-ddos-attack

[101] M. Vasek, M. Thornton, and T. Moore, "Empirical analysis of denial-of-service attacks in the bitcoin ecosystem," in *Proc. Int. Conf. Financial Cryptography Data Secur.*, 2014, pp. 57–71.
[102] M. Mirkin, Y. Ji, J. Pang, A. Klages-Mundt, I. Eyal, and A. Juels, "BDoS: Blockchain denial-of-service," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2020, pp. 601–619.
[103] M. Ahmed, J. Wei, Y. Wang, and E. Al-Shaer, "A poisoning attack against cryptocurrency mining pools," in *Proc. Int. Workshop Cryptocurrencies Blockchain Technol.*, 2018, pp. 140–154.
[104] Y. Wang, J. Yang, T. Li, F. Zhu, and X. Zhou, "Anti-dust: A method for identifying and preventing Blockchain's dust attacks," in *Proc. Int. Conf. Inf. Syst. Comput. Aided Educ. (ICISCAE)*, Jul. 2018, pp. 274–280.
[105] D. Rhodes. (2018). *51% Attack Security: Delayed Proof of Work (dPoW)*. [Online]. Available: https://blog.komodoplatform.com/en/delayed-proof-of-work
[106] K. Liao and J. Katz, "Incentivizing double-spend collusion in bitcoin," in *Proc. Financial Cryptography Data Secur.*, 2017, pp. 1–16.
[107] R. Alcarria, B. Bordel, T. Robles, D. Martín, and M.-Á. Manso-Callejo, "A blockchain-based authorization system for trustworthy resource monitoring and trading in smart communities," *Sensors*, vol. 18, no. 10, p. 3561, Oct. 2018.
[108] R. Yuan, Y.-B. Xia, H.-B. Chen, B.-Y. Zang, and J. Xie, "ShadowEth: Private smart contract on public blockchain," *J. Comput. Sci. Technol.*, vol. 33, no. 3, pp. 542–556, May 2018.
[109] G. Bissias, B. N. Levine, A. P. Ozisik, and G. Andresen, "An analysis of attacks on blockchain consensus," 2016, *arXiv:1610.07985*. [Online]. Available: http://arxiv.org/abs/1610.07985
[110] S. Henningsen, D. Teunis, M. Florian, and B. Scheuermann, "Eclipsing ethereum peers with false friends," in *Proc. IEEE Eur. Symp. Secur. Privacy Workshops (EuroS PW)*, Jun. 2019, pp. 300–309.
[111] P. Ekparinya, V. Gramoli, and G. Jourjon, "Double-spending risk quantification in private, consortium and public ethereum blockchains," 2018, *arXiv:1805.05004*. [Online]. Available: http://arxiv.org/abs/1805.05004
[112] C. Grunspan and R. Pérez-Marco, "Double spend races," 2017, *arXiv:1702.02867*. [Online]. Available: http://arxiv.org/abs/1702.02867
[113] J. Jang and H. N. Lee, "Profitable double-spending attacks," *J. Appl. Sci.*, vol. 10, pp. 1–23, Jan. 2020.
[114] G. O. Karame, E. Androulaki, M. Roeschlin, A. Gervais, and S. Čapkun, "Misbehavior in bitcoin: A study of double-spending and accountability," *ACM Trans. Inf. Syst. Secur.*, vol. 18, no. 1, pp. 1–32, Jun. 2015.
[115] Y. Sompolinsky and A. Zohar, "Bitcoin's security model revisited," 2016, *arXiv:1605.09193*. [Online]. Available: http://arxiv.org/abs/1605.09193
[116] G. Ramezan, C. Leung, and Z. Jane Wang, "A strong adaptive, strategic double-spending attack on blockchains," in *Proc. IEEE Int. Conf. Internet Things (iThings), IEEE Green Comput. Commun. (GreenCom), IEEE Cyber, Phys. Social Comput. (CPSCom), IEEE Smart Data (SmartData)*, Jul. 2018, pp. 1219–1227.
[117] I. Osipkov, E. Y. Vasserman, N. Hopper, and Y. Kim, "Combating double-spending using cooperative P2P systems," in *Proc. 27th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2007, pp. 41–50.
[118] A. P. Ozisik and B. N. Levine, "An explanation of Nakamoto's analysis of double-spend attacks," 2017, *arXiv:1701.03977*. [Online]. Available: http://arxiv.org/abs/1701.03977
[119] J. P. Podolanko, J. Ming, and M. Wright, "Countering double-spend attacks on bitcoin fast-pay transactions," in *Proc. IEEE Comput. Society's Tech. Committee Secur. Privacy*, Apr. 2017, pp. 1–3.
[120] M. Grundmann, T. Neudecker, and H. Hartenstein, "Exploiting transaction accumulation and double spends for topology inference in Bitcoin," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, 2019, pp. 113–126.
[121] W. Xiaoding, S. Garg, H. Lin, M. Jalilpiran, J. Hu, and M. S. Hossain, "Enabling secure authentication in industrial iot with transfer learning empowered blockchain," *IEEE Trans. Ind. Informat.*, early access, Jan. 5, 2021, doi: 10.1109/TII.2021.3049405.
[122] R. Sagar, R. Jhaveri, and C. Borrego, "Applications in security and evasions in machine learning: A survey," *Electronics*, vol. 9, no. 1, p. 97, 2020.
[123] S. Rani, D. Gupta, S. Garg, M. Jalilpiran, and M. S. Hossain, "Consumer electronic devices: Evolution and edge security solutions," *IEEE Consum. Electron. Mag.*, early access, Mar. 3, 2021, doi: 10.1109/MCE.2021.3062800.
[124] M. Numan, F. Subhan, W. Z. Khan, S. Hakak, S. Haider, G. T. Reddy, A. Jolfaei, and M. Alazab, "A systematic review on clone node detection in static wireless sensor networks," *IEEE Access*, vol. 8, pp. 65450–65461, 2020.
[125] M. Alazab and R. Broadhurst, "An analysis of the nature of spam as cybercrime," in *Cyber-Physical Security*, R. Clark, S. Hakim, Eds. Cham, Switzerland: Springer, 2017, pp. 251–266.
[126] S. R. K. Somayaji, M. Alazab, M. Mk, A. Bucchiarone, C. L. Chowdhary, and T. R. Gadekallu, "A framework for prediction and storage of battery life in IoT devices using DNN and blockchain," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2020, pp. 1–6.
[127] A. Bhardwaj, S. B. H. Shah, A. Shankar, M. Alazab, M. Kumar, and T. R. Gadekallu, "Penetration testing framework for smart contract blockchain," *Peer Peer Netw. Appl.*, vol. 13, pp. 1–16, Sep. 2020.
[128] S. Tikhomirov, E. Voskresenskaya, I. Ivanitskiy, R. Takhaviev, E. Marchenko, and Y. Alexandrov, "SmartCheck: Static analysis of ethereum smart contracts," in *Proc. 1st Int. Workshop Emerg. Trends Softw. Eng. Blockchain*, May 2018, pp. 9–16.
[129] D. Qin, C. Wang, and Y. Jiang, "RPchain: A blockchain-based academic social networking service for credible reputation building," in *Proc. Int. Conf. Blockchain*, 2018, pp. 19–183.
[130] M. Saad, L. Njilla, C. Kamhoua, and A. Mohaisen, "Countering selfish mining in blockchains," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Feb. 2019, pp. 360–364.
[131] S. Gajbhiye, S. Karmakar, M. Sharma, and S. Sharma, "Paradigm shift from classical cryptography to quantum cryptography," in *Proc. Int. Conf. Intell. Sustain. Syst. (ICISS)*, Dec. 2017, pp. 548–555.
[132] J. Moubarak, E. Filiol, and M. Chamoun, "On blockchain security and relevant attacks," in *Proc. IEEE Middle East North Afr. Commun. Conf. (MENACOMM)*, Apr. 2018, pp. 1–6.
[133] O. Boireau, "Securing the blockchain against hackers," *Netw. Secur.*, vol. 2018, no. 1, pp. 8–11, Jan. 2018.
[134] V. Saini. (2020). *How to Write Upgradable Smart Contracts*. [Online]. Available: https://simpleaswater.com/upgradable-smart-contracts
[135] M. Tena. (2017). *7 Regulatory Challenges Facing Blockchain*. [Online]. Available: https://www.bbva.com/en/7-regulatory-challenges-facing-blockchain
[136] G. Manogaran, M. Alazab, P. M. Shakeel, and C.-H. Hsu, "Blockchain assisted secure data sharing model for Internet of Things based smart industries," *IEEE Trans. Rel.*, early access, Feb. 8, 2021, doi: 10.1109/TR.2020.3047833.

MUBASHAR IQBAL received the B.S. degree in computer science and the M.S. degree in human–computer interaction. He is currently pursuing the Ph.D. degree with the University of Tartu (UT), Estonia. He has been working as a Junior Researcher with UT, since 2019. He is currently involved in the ERASMUS+ Sectoral Alliance Program CHAISE. His research interests include the security implications of blockchain systems and the implementation of a security risk management framework for blockchain systems, concentrating specifically on the security of blockchain-based decentralized applications.

RAIMUNDAS MATULEVIČIUS received the Ph.D. Diploma degree in computer and information science from the Norwegian University of Science and Technology. He currently holds a Professor of information security position at the University of Tartu, Estonia. His publication record includes more than 100 articles published in peer-reviewed journals, conferences, and workshops. He is the author of a book *Fundamentals of Secure System Modelling* (Springer, 2017). He is involved in the SPARTA H2020 Project (task: Privacy-by-Design), ERASMUS+ Sectoral Alliance Program CHAISE. He is also a Principal Researcher in a few other international and national projects. His research interests include security and privacy of information, security risk management, and model-driven security. He has been a Program Committee Member at international conferences (e.g., NordSec, PoEM, REFSQ, and CAiSE).