The Double Spending Dilemma: From Traditional Cash Systems to Interoperable Blockchain Systems

Aradhita Mukherjee[3], Luca Olivieri[1], Nabendu Chaki[2] and Agostino Cortesi[1]

[1]*Ca' Foscari University of Venice, Venice, Italy*
[2]*University of Calcutta, Kolkata, India*
[3]*Netaji Subhash Engineering College, Kolkata, India*

Abstract

Digital cash systems have been addressing the double spending problem for decades. Unlike the physical world, where currency naturally prevents duplication, digital systems must implement complex mechanisms to avoid double spending. In this positional paper, we examine the evolution of the double spending issue from early digital cash models to modern blockchain-based solutions, focusing also the emergence of double spending risks in interoperable blockchain ecosystems. We explore how the rise of blockchain interoperability introduces new vulnerabilities, potentially enabling novel forms of double spending across interconnected networks. In particular, we show how differences in economic conditions across blockchains, inconsistencies in network timing and transaction finality, market manipulations, speculative behaviors, and weaknesses in cross-chain mechanisms can collectively create new opportunities for double spending. By illustrating these issues through practical examples, we demonstrate that interconnected blockchain ecosystems require security models that extend beyond the assumptions of isolated networks.

Keywords

Double-spending, payment systems, digital currency, cryptocurrency, blockchain, distributed ledger technology, consensus mechanism, blockchain interoperability, cross-chain, cross-blockchain, multi-chain, multi-blockchain.

1. Introduction

Double spending is a well-known problem in physical and digital cash systems. Before the advent of fully automated and digital systems, double-spending controls were slow and vulnerable: verifying funds took time, signatures and checks could be easily forged, and fraudulent transactions could go unnoticed for days, due to manual paper-based investigation. For instance, as depicted in the movie "Catch Me If You Can" (2002), which is based on the real-life exploits of Frank Abagnale Jr., one of his many clever schemes involves effectively double spending using bank checks. The following is a breakdown of how he carries out this scam:

- Frank poses as a Pan Am pilot. This gets him credibility and access to airline facilities. It also means banks treat his checks from Pan Am with trust.
- He creates fake payroll checks. Using his printing skills, he forges Pan Am payroll checks, he makes them look like legit paychecks.
- He deposits part of the money from a check into his bank account. Then he asks for part of it in cash immediately (this was more common in the '60s banking system). Since the check takes time to clear, the bank gives him money assuming it's legit (double cashing).
- The sneaky genius part is the drop box trick: he places fake checks into the airport's night deposit drop box for Pan Am's payroll account. The bank processes them as if Pan Am dropped them off. The checks look real and come from a "trusted" source, no red flags go off.

Joint National Conference on Cybersecurity (ITASEC & SERICS 2026), February 9-13, 2026, Cagliari, IT
aradhita.mukherjee@unive.it (A. Mukherjee); luca.olivieri@unive.it (L. Olivieri); nabendu@ieee.org (N. Chaki); cortesi@unive.it (A. Cortesi)
0000-0001-8787-9086 (A. Mukherjee); 0000-0001-8074-8980 (L. Olivieri); 0000-0003-3242-680X (N. Chaki); 0000-0002-0946-5440 (A. Cortesi)

- He replicates this multiple times. With various banks and small check amounts that stay under fraud radar.

Why It Worked?

- Banks in the '60s did not have the instant verification systems we have today.
- A good-looking check from a known company + a convincing person = almost guaranteed cash.
- He exploited the float time (the delay between check deposit and processing).

Essentially, Frank created fake value (counterfeit checks); got cash plus deposits for the same fake funds; finally, he spent money that did not exist, and moved on before the bank realized. In summary, this story allows us to clearly identify the aspects that are relevant for double spending: trust mechanisms, access control mechanisms, event synchronization, and the presence or absence of monitoring and control tools. Notably, similar vulnerabilities can still occur even in traditional digital systems, particularly because they still required trusted intermediaries and centralized systems for verification and validation. However, with the advent of blockchain technology, the paradigm has shifted from authority-based solutions to trustless environments that discourage double spending through cryptographic and consensus-based guarantees.

Although blockchain technology has been a killer application in this context, as blockchain systems evolve towards interoperability, the challenge becomes more complex. Ensuring seamless and secure communication across heterogeneous networks introduces new attack surfaces, potentially enabling double spending across interconnected chains. At the current state of the art, although previous studies have examined blockchain interoperability and its general challenges, there remains a lack of systematic investigation into how interoperability mechanisms might reintroduce double spending vulnerabilities across interconnected blockchains. Most existing research focuses on interoperability protocols or bridge architectures but seldom classifies the distinct layers where double-spending threats can occur. In this paper, we provide the following contributions:

- a systematic overview of how the concept of double spending evolved from traditional authority-based systems to trustless digital environments that relied on blockchain technology;
- an investigation of open issues related to blockchain interoperability, focusing on the double-spending issues.
- practical scenarios illustrating how economic disparities between chains, timing and finality inconsistencies, and weaknesses in cross-chain mechanisms and smart contracts can enable new forms of double spending.

Paper Structure Section 2 introduces the double spending problem. Section 3 and Section 4 describe solutions for double spending adopted by traditional cash systems and blockchain systems, respectively. Section 5 deals with double spending in the context of blockchain interoperability. Finally, Section 6 concludes the paper.

2. The Double Spending Problem

Spending is *"the act of giving money for goods and services"*.[1] In this case, *"giving"* results in a form of loss because the giver no longer has control or possession of what they gave away. Specifically, in the physical world, spending money (or more precisely *currency*) means transferring the ownership of one or more banknotes and coins, i.e. physical objects, from one actor to another. Once these objects are transferred, the sender can no longer use them, as they are physically moved to the receiver. In other words, the same banknote or coin is spent only once at a time and cannot exist in two places at the same time. However, with the advent of the digital era, the concept of currency has evolved to include digital forms and payments. Digital currency is no longer physical but a non-tangible digital data item. Hence, this raises an important question:

[1]Definition from Cambridge Dictionary (https://dictionary.cambridge.org/dictionary/english/spending)

How to guarantee that the same digital currency is spent only once and not two or more times?

This problem is also known as *double-spending* and happens when the same digital currency could be spent multiple times by the same sender before receivers can detect duplication. The consequences can lead to financial losses, monetary devaluation and system collapse due to loss of credibility. It also enables fraud, as recipients can release goods or services believing they have been paid for, only to later discover that the transaction was invalid and they therefore did not receive payment.

3. Authority-based Solutions for Double Spending

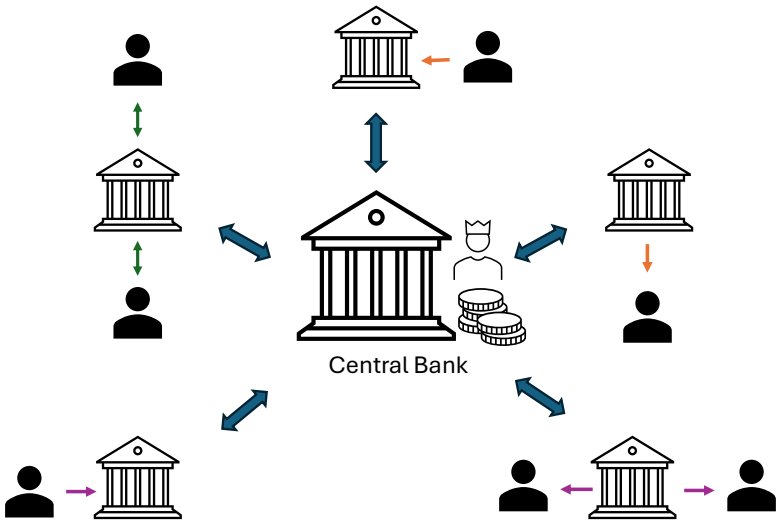

Figure 1: Traditional Authority-based Bank Architecture

In traditional digital cash systems, the guarantees against double spending are usually provided by banks, payment processors (e.g., *Mastercard*, *Visa*, *Amex*, …), and other affirmed financial institutions. This means that the entire system is based on the trust of third parties. For instance, as shown in Figure 1, a simplified version of bank paradigm. Banks record their customers' transactions in their own internal systems, but the actual transfer of money between institutions occurs in the central bank's ledger, where the reserve accounts of commercial banks are held. When a payment moves from one bank to another, the central bank updates these accounts through its payment system, which serves as the official register of interbank movements and ensures the security and reliability of the entire financial system. These intermediaries ensure the legitimacy of transactions, maintain transaction records, and handle disputes. People generally trust banks and payment processors to prevent issues like double spending because these institutions operate under strict regulations and typically apply secure and reliable systems to track and verify transactions and the exchange of digital currencies. Then, trust is centralized under the control of these entities, which prevents double spending by maintaining a central ledger that tracks all transactions and ensures that once currency is spent, it cannot be used again.

Their long-standing presence, government backing, and ability to offer customer support make traditional digital cash systems seem safe and dependable. However, this trust is not absolute and has been questioned over time, as in the example of Section 1. Indeed, although they are highly automated, their management is carried out by humans and institutions who can make mistakes, act maliciously or be subject to cyber attacks [1] and failures [2, 3]. Furthermore, they can set arbitrary fees and prices (currency exchange fees, international transfer fees, …), requiring also long verification times such as several hours or days to ensure the large amounts of currency or the transfer of high-value economic assets.

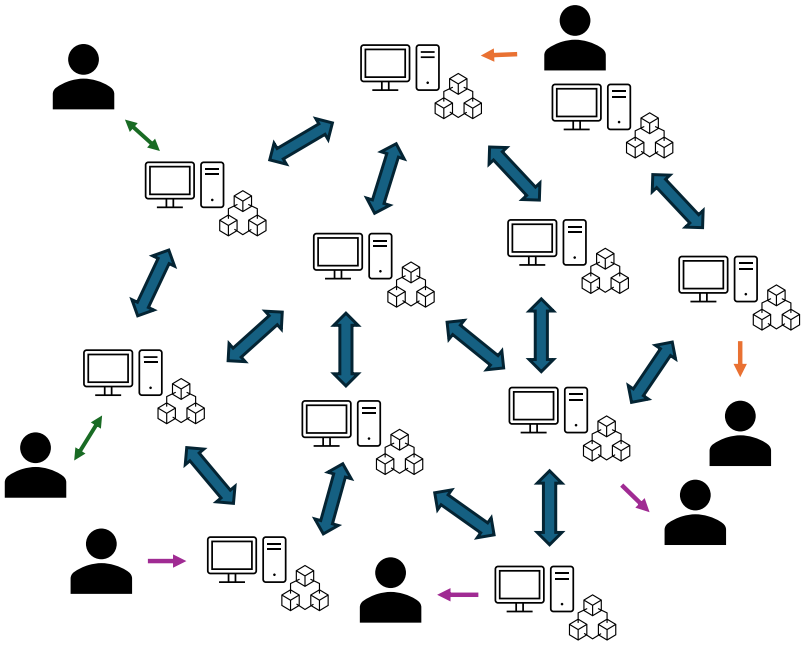

Figure 2: Distributed and Decentralized Blockchain Architecture

Around the '90s, numerous alternative solutions to the double-spending problem were proposed, some theoretical and others implemented with modest success. Though none achieved widespread adoption or fully resolved the issue, each played a role in shaping future advancements. For instance, Chaum [4] proposed a notable crypto system by using RSA-based blind signatures, later known as *eCash*, to enable users to store digital currency on their personal computers and avoid double spending. Despite its innovation, eCash faced a major hurdle: its dependence on banks. This reliance created a single point of failure, as users had to notify their bank before depositing eCash. There was no guarantee that a recipient's bank would accept the digital currency, and without a willing institution, the funds became effectively unusable.

4. Blockchain: a Trustless Solution for Double Spending

While traditional digital cash systems remain deeply integrated into everyday life, their role is increasingly challenged by technologies that promise greater control, transparency and independence. For instance, Back [5] introduced a Proof-of-Work algorithm within *HashCash*. Despite its name, HashCash did not involve the exchange of digital currency, but it was focused on mitigate denial-of-service attacks and reduce email spam. However, it contributed a key innovation to double-spending prevention that would later influence the development of other digital cash such as b-money [6] and Bit Gold [7]. Unlike eCash, HashCash aimed to address the double-spending problem without relying on a central authority. Furthermore, it is also the precursor of Bitcoin [8], the first mainstream digital cash system based on blockchain, born during the economic and financial crisis of the early 2000s (The Great Recession) [9].

Blockchain technology has become disruptive for alternative digital transfer payments [8, 10] thanks to cost reductions, transaction speed up, the elimination of intermediaries, and the increment of financial inclusion.

Regarding double spending, it is commonly asserted that blockchain technology inherently solves the problem. However, a more accurate characterization is that **blockchain strongly mitigates the risk of double spending rather than eliminating it entirely**, thanks to a *trustless* environment[2]

[2]A trustless environment refers to a system where participants do not necessarily trust each other, but this does not imply that the environment consists of untrusted participants only.

and mechanisms that make double spending disadvantageous on different sides (i.e. computer science, social and economic).

The blockchain is a ledger technology with an abstract data structure shared in redundant copies among a decentralized and distributed network (see Figure 2). Transaction data is collected into blocks, each cryptographically linked to the previous one using one-way hash functions that make it computationally hard to reverse the cryptographic process, which means making tampering attempts evident, such as the tampering for double spending purposes. Furthermore, since the data structure is shared in redundant copies, the network can detect tampering, as subsequent blocks would no longer match. In addition, the data structure is visible to all participants (unless specific settings of access control adoptable by *permissioned* blockchains), who can report any anomalies to the blockchain communities to ensure that measures are taken in this regard. The *consensus mechanism* is the component that manages the blockchain data structure. It allows network participants to agree on the validity and consistency of transactions, adding new records within the data collection, preventing unauthorized changes, ensuring trust among decentralized parties, as well as preventing the duplication of digital currency (aka *cryptocurrency* in the blockchain context) and mitigating double spending issues through a reward and penalty system that makes cheating inconvenient. The consensus mechanism typically relies on majority agreement and incorporates properties similar to Byzantine Fault Tolerance.[3] For instance, considering the current most widely used consensus mechanisms: *Proof-of-Work* (PoW) and *Proof-of-Stake* (PoS).

In PoW, the consensus is implemented by *miners*, network peers required to solve complex computational puzzles, to propose a candidate block that may be added to the ledger. If the candidate block is successfully validated by the majority of the network, it is added to the blockchain, and the miner is typically rewarded with transaction fees and, depending on the blockchain, potentially also with new cryptocurrency minted. Otherwise, the block is discarded and there is no reward, that implies lose the resources spent (e.g. energy consumption, hardware deterioration, ...) to compute the puzzle, which will no longer be valid for other block candidacy. Therefore, adding a double spend in the candidate block could lead to a failure of validation by the network, resulting in economic damage to the cheater.

In PoS, the consensus is based on peers called *validators*, a subset of the network that freezes an amount of cryptocurrencies called *stake* as a guarantee for validations. In this case, there are no miners and puzzles to solve. PoS elects validators based on a combination of factors such as stake size and randomness, distributing the chances to be selected for the validation among multiple participants. Ideally, the validator with a higher stake than the others has a higher chance of being chosen for validation. That means, the validator has more chance to receive the reward because it is issued when a block is successfully validated, confirmed by other validators and added to the blockchain. However, on the other hand, it has also a greater risk of losing the stakes, because in the event that the transaction is not successfully validated, the validator suffers a penalty which consists in the partial or total reduction of its stake (aka *stake slashing*). In this case, a malicious validator is discouraged to cheat during validation because this leads to the loss of the reward but also to the reduction of the stakes, as well as the chances of validation.

Although double spending is primarily mitigated in blockchain through the combination of the tamper-proof data structure and the decentralized consensus mechanism, a minimal or theoretical risk still exists, even on major and well-established networks. For instance, this could occur in *51% attack* [12], *Finney attack* [13], *Eclipse attack* [14] and other attack scenarios [15, 16, 17]. Furthermore, other numerous studies have examined various double-spending attack strategies [18, 19, 20, 21, 22].

5. Blockchain Interoperability and Double Spending

At the beginning of blockchain technology, numerous implementations have emerged in isolation, each designed for specific use cases or ecosystems. Over time, this fragmented development has led to a landscape of disconnected or hermetic networks. For this reason, blockchain interoperability has

[3]The ability of a system to keep working even if some nodes fail or act maliciously [11]

emerged as a critical requirement to facilitate exchanges and transfers between different ecosystems, improving transparency in inter-chain communications and potentially coordinating international regulatory frameworks, contracts, and agreements [23, 24]. However, the literature on the blockchain interoperability is still limited [25, 26, 27], with only a few more specific studies on prevention and possible attack vectors [28, 29], and there is also a lack of studies on adequate verification tools [30, 31]. For instance, there is an urgent need for robust finality protocols and enhanced state-consistency safeguards in Layer-2 environments [32]. Notland et al. [33] underscored the importance of modular, formally verified bridge designs with provable security guarantees. Bappy et al. [34] highlight the importance of real-time frameworks for resolving transaction conflicts. In addition, another promising approach involves integrating AI-driven anomaly detection into blockchain infrastructures, enabling the early identification of threats [35].

Regarding double spending, Section 4 states that blockchain *"solves"* the problem, but this is true only when considering a single blockchain. In fact, in contexts with multiple blockchains, the promises on double spending could fail[36]. Below, we outline some of the key pitfalls and issues in multi-chain systems.

5.1. Consensus layer and Economic Disparities Between Blockchains

At the consensus layer, risks arise from inconsistencies in block finality or reorganization events, such as 51% attacks or long-range reorgs, which may propagate through interconnected networks and create divergent transaction histories. Although theoretically possible, coordinating such attacks across multiple blockchains with distinct consensus mechanisms poses significant technical and economic challenges. The need for synchronized control, combined with the presence of observer networks and fraud-detection systems, greatly reduces the likelihood of large-scale exploitation.

However, a double spending due to a rational consensus may still emerge in asymmetric interoperability scenarios involving minor and consolidated blockchains. Indeed, the most important feature of blockchains to prevent double spending from occurring is the concept that cheating is highly harmful in socio-economic terms. In individual blockchains, the attacker is discouraged from committing double spending, both due to the consensus and cryptography mechanisms, which are driven and maintained mainly by the risk of financial loss or a decline in its value. However, in interoperability settings, blockchains may vary in their economic and social environments due to differences in token value, transaction fees, and reward mechanisms. This means that could lead to a disparity. It might undermine the socio-economic balance that typically discourages malicious behaviour in the blockchain system and consequently leads to increased risks of fraud across interconnected blockchains. For instance, consider two blockchains A and B, where A is significantly more profitable than B. In this scenario, all peers operating on B may decide to exploit double spending attacks. Their goal is to extract as much value as possible, both in terms of digital assets and goods purchased, before the compromise of B's integrity is detected by users of A. These actors may also hold accounts on the more stable and valuable blockchain A, allowing them to transfer the extracted assets for additional gain. Although A is designed to be immutable and resistant to rollbacks, this immutability ironically becomes a vulnerability. Once fraudulent transactions are confirmed, they cannot be reversed without a majority, which is typically difficult to obtain if the fraud involves only a few wealthy users of the blockchain A.

Then, disparities in token value, transaction fees, or reward structures can disrupt the socio-economic balance that normally deters attacks, allowing actors from weaker networks to exploit cross-chain links for profit through double-spending strategies [36].

5.2. Synchronization and Confirmation-Time Discrepancies

Confirmation time refers to the period during which a transaction is considered final and irreversible on a blockchain. This is typically measured by the number of blocks added after a transaction is included in a block, making it increasingly difficult to alter or reverse the transaction, i.e., to reduce the risk of chain reorganizations and double-spending attacks. It is essential in scenarios like the release of goods,

where ensuring that payment or transaction confirmation is secure before goods are delivered to the buyer is crucial.

However, different blockchains operate on independent consensus mechanisms and finality models, which leads to variations in how quickly transactions are confirmed and considered irreversible. Typically, blockchains based on consensus mechanism like PoW require multiple block confirmations, often six or more, to consider a transaction final. For instance, in Bitcoin, this process can take around 60 minutes or longer due to the average 10-minute block time. In contrast, in PoS consensus mechanisms such as Ethereum 2.0, Cosmos, or Polkadot typically achieve finality within seconds to a few minutes, depending on the specific protocol.

These discrepancies can lead to timing mismatches during cross-chain operations. A faster blockchain may treat a transaction as final, while a slower blockchain is still processing or awaiting confirmations. This asynchronous behavior introduces risks, such as premature actions based on unconfirmed states or exposure to chain reorganizations before finality is achieved. An attacker could exploit this by initiating a transaction on the slower chain while simultaneously executing a conflicting transaction on the faster one. If the slower chain has not yet confirmed the transaction, the attacker may successfully double spend.

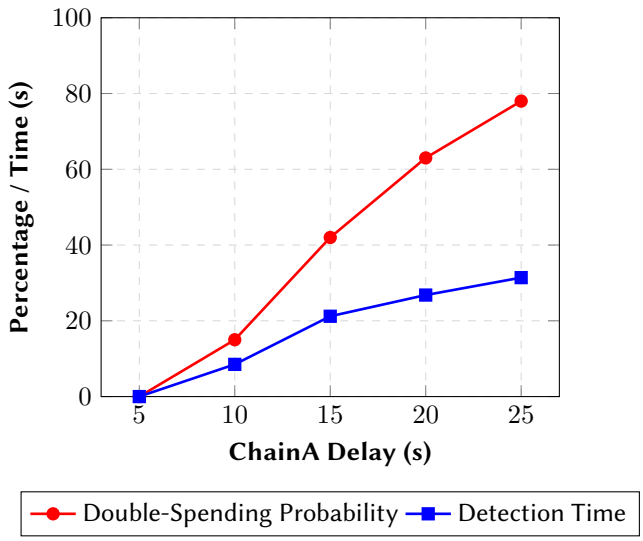

Figure 3: Impact of ChainA Delay on Double-Spending Probability and Detection Time.

Figure 3 shows an illustrative experiment[4] we conducted between two Ethereum-protocol-based blockchains, *ChainA* and *ChainB*, to evaluate the impact of confirmation delays on two critical performance metrics: the probability of double spending and the corresponding detection time. The experimental results reveal a strong correlation between network delay and the overall reliability of inter-chain operations. When the delay in *ChainA* remains low, the interoperability framework maintains atomicity and consistency, ensuring that transactions are securely validated across chains. However, as the delay increases, the probability of double-spending grows sharply. This escalation demonstrates that delayed confirmations disrupt the temporal alignment required for consistent cross-chain state updates, allowing adversaries to exploit asynchronous transaction windows.

In multi-chain ecosystems, interoperability depends not only on protocol compatibility but also on precise temporal coordination. The interoperability layer must dynamically handle latency variations to maintain atomic cross-chain commitments and timely fraud detection. Techniques such as

[4]The experiment was carried out on an Ubuntu 24.04 LTS (64-bit) system equipped with 16 GB RAM and an Intel Core i7 processor (2.81 GHz). The experimental environment utilized Ganache and tools from the Truffle Suite to design, simulate, and debug Ethereum-based blockchains, including smart contracts, user accounts, miners, and tokens. To facilitate cross-chain interactions, Node.js and Web3.js were employed to develop a bridge between two independent blockchain networks, thereby emulating realistic cross-chain communication and interoperability scenarios. Code available at: https://github.com/aradhita1988/Double-Spending-Dilemma-Conference

asynchronous consensus anchoring, timestamp-based verification, and cross-chain finality proofs help reduce delay-induced inconsistencies. Overall, interoperability in multi-chain systems is inherently delay-sensitive—unmanaged latency shifts from a performance concern to a security threat. Thus, practical frameworks must emphasize low-latency synchronization, resilient state validation, and adaptive trust management to ensure secure and efficient cross-chain operations.

5.3. Token Transfer and Bridge-based Trustness

Inter-blockchain communication is a challenging task because first-generation blockchains like Bitcoin and Ethereum were not designed with interoperability in mind. Therefore, they lack native protocols for these purposes. In contrast, more recent blockchain platforms such as Cosmos [37] and Polkadot [38] prioritise interoperability, but mainly within their own ecosystems. As a result, external solutions such as *bridges* are required to enable communication and asset transfers between different blockchains, implementing different token transfer mechanisms [39].

However, the adoption of bridges raises concerns because they are external solutions that often require trust, specifically trust in the bridge's design and the integrity of its operators, opening the window to double spending opportunities [40]. Bridge trustness often breaks down into two key comparisons [39]:

- **Custodial vs. non-custodial**. In a custodial bridge, a third party holds custody of the user's currency. In this model, users have no choice but to trust that the custodian will act honestly, maintain proper reserves, and allow withdrawals when requested. This custodial setup can be fast and efficient, but it is still trusted-based, because one have to trust whoever manages it and the code that is executed, which is not necessarily available. On the other hand, a non-custodial bridge removes the need to trust a central asset holder. Instead, these bridges use smart contracts or decentralized validators to handle the transaction without relying on any single party. Non-custodial bridges are generally seen as more secure in theory, because they remove the human element from asset custody. However, this also means the security depends entirely on the quality of the smart contracts and the reliability of the network. If a contract is exploited or if validators collude, double spending still possible.
- **Centralized vs. decentralized**. A centralized bridge is run by a single organization or a small group of operators, typically off-chain. These entities are responsible for validating transactions, managing the infrastructure, and ensuring that currency is correctly exchanged between blockchains. This model is often favored for its speed and simplicity. However, this poses the same problems as traditional digital cash systems, i.e. it is necessary to trust third parties (see Section 3). If the bridge's operators are hacked, act maliciously, or become subject to social-economic pressure, it could lead to double spending issues. Furthermore, the centralized model introduces also a single point of failure. In contrast, a decentralized bridge distributes responsibility across multiple independent validators, smart contracts, or consensus mechanisms. Decentralized bridges are typically more transparent, since transactions and validator behavior are handled on-chain. However, decentralization adds complexity and longer times for validation. In this model, users shift from trusting people to trusting code and game-theoretic incentives, which can be equally dangerous in the case of poor system design or buggy code implementations. Furthermore, several decentralized bridges rely on governance mechanisms to manage upgrades, pause operations, or validate transfers. In many designs, this governance is enforced through multi signature wallets that control critical contracts or pooled funds. In this context, a *governance attack* can occur if the entities controlling these *multi-sig* keys are compromised, collude, or act maliciously. An attacker who gains control of a sufficient number of keys could approve fraudulent transfers, deploy malicious contract upgrades, drain locked funds, or even disable the bridge entirely, effectively enabling double spending. This risk is particularly pronounced because several bridges are custodial or semi custodial in practice, even when they are marketed as fully decentralized.

Beyond these clear distinctions, *hybrid models* have emerged that blend different design approaches to mitigate their respective drawbacks. One example is the *federated bridge*, which relies on a trusted group of validators (i.e., a *federation*), rather than a fully decentralized or trustless system. In this model, a select set of known entities is responsible for validating and approving cross-chain transactions.

There are also alternative approaches to bridges, such as *atomic swap* [41], that enable two parties to directly exchange assets across blockchains using cryptographic primitives such as *hash time locked contracts* (HTLC) [41]. If implemented correctly, these contracts can guarantee an atomicity: either both sides of the exchange are executed, or neither is. Then, two users can interact without necessarily relying on a bridge, although bridges also may use this type of contract for exchanges [39]. The difference lies in the automation, the speed of transfer and the interactions of the parties involved. It depends, in fact, on adequate liquidity and the presence of active counterparties, who if they do not confirm some operations in time could invalidate the exchange. Furthermore, HTLC contracts have their limitations. They generally require compatible scripting capabilities on both the source and target blockchains, such as common hashing primitives and sufficiently expressive programming features. They can be less flexible for complex cross-chain interactions and do not easily support generalized message passing or arbitrary smart contract calls. In addition, preventing double spending may require each involved party to independently verify the transaction state, adding operational overhead.

Nonetheless, according to Schulte et al. [25], the fundamental challenge remains: it is inherently difficult to fully replicate the state of one blockchain within another blockchain and verify that non-trial change events occur in both without really relying on a third party.

5.4. Market Manipulation and Speculative Behaviors

Exchanging tokens on the blockchain is not instant, it takes time. During this time a token can lose or gain value, as well as the transaction fees associated with it. This phenomenon can become even more accentuated in the context of interoperability where multiple blockchains and ecosystems are involved. Consequently, a token can increase in value immediately after a transaction or exchange. While this is not classic double spending, it reflects market dynamics driven by supply and demand in non stable assets. However, a token value can also be partially or significantly influenced in different ways by actors involved in blockchain environments, effectively making the loss or gain of value to the detriment of some users intentional.

For instance, one form of widespread speculative behavior is reflected in the issue of *Maximal Extractable Value* (MEV) [42, 43, 44], where participants can strategically reorder, include, or exclude transactions within a block to extract additional profit (e.g., *front-running*, *back-running*, and *sandwich* attacks), reducing market fairness and disadvantages ordinary participants. In blockchain interoperability settings, MEV may become more complex because it can arise not only within a single blockchain but also between multiple blockchains. Actors may exploit price discrepancies between decentralized exchanges on different networks, anticipate bridge transfers, or manipulate the timing of cross-blockchain messages to capture arbitrage opportunities.

Another critical issue concerns the reliance of bridges and other cross chain exchange mechanisms on *oracles* [45]. These systems depend on external price feeds to compute transaction fees, determine conversion rates, and validate asset equivalence across networks. As a result, the correctness and security of cross-blockchain operations are directly tied to the integrity of oracle data. The so called *oracle problem* is pervasive because oracles operate at the boundary between on-chain and off-chain environments, where guarantees of decentralization and verifiability are weaker. Price feeds can be manipulated through market attacks, data corruption, collusion among oracle operators, or delays in data updates. Moreover, achieving strong trust assumptions is inherently difficult, since users must rely on the accuracy, availability, and honesty of external data providers. Ensuring reliable oracle inputs remains one of the central challenges in decentralized and interoperable blockchain infrastructures. Ensuring reliable oracle inputs remains one of the central challenges in decentralized and interoperable blockchain infrastructures, although possible solutions and mitigation strategies for the problem have been worked on in recent years [46, 47, 48, 49, 50, 51].

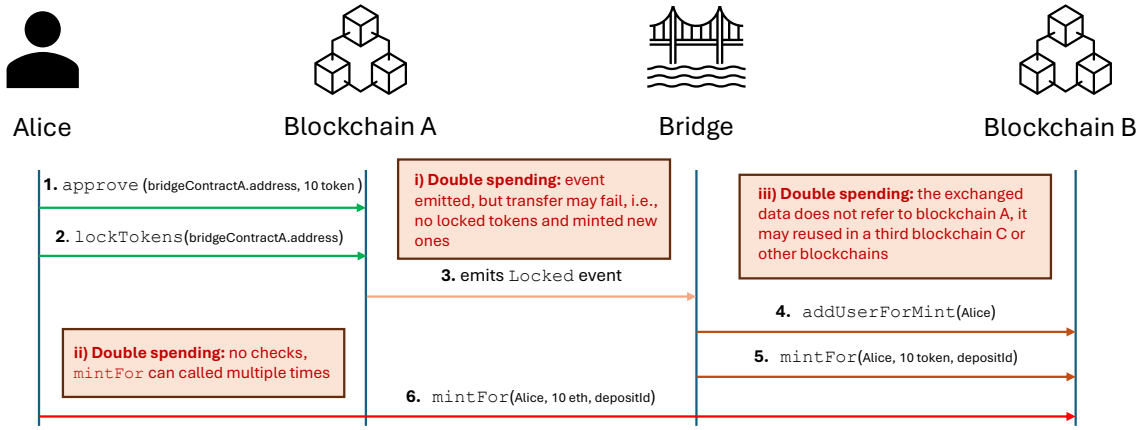

Figure 4: Double-spending issue due to bugs in the smart contracts/ application layer.

5.5. Security Gaps and Exploits

Developing cross-chain protocols and external solutions like bridges is inherently error-prone due to their complexity, lack of standardization, and absence of well-established design patterns. Currently, the ecosystems rely heavily on custom, ad-hoc implementations crafted by practitioners, rather than on standardized frameworks or reusable architectures. The lack of a unified approach increases the likelihood of design flaws, integration bugs, and security vulnerabilities, particularly in critical components such as message validators, and asset locking mechanisms.

This problem is also reflected in the current state of verification and security tools, which are primarily focused on individual blockchains and smart contracts [30]. While there are many mature tools available for analyzing and verifying individual components, such as static analyzers, formal verification frameworks, and runtime monitors, there are very few equipped to handle cross-chain interactions [52, 53, 54, 55, 56, 57]. The complexity and lack of standardization in interoperability protocols make it difficult to model and verify behaviors that span multiple blockchains [31]. As a result, critical components like bridges and cross-chain message handlers are often left unverified or only informally tested, exposing them to subtle logic flaws and coordinated attacks. Just consider that, at the end of 2024, more than $2.8 billion dollars has been lost due to cross-chain exploits [58, 39, 59], also including double spending (e.g. the exploit to Polygon Plasma Bridge [60, 61]).

At the application layer, vulnerabilities stem from inconsistencies in smart contract logic, oracle dependencies, or state replication across chains. In decentralized finance (DeFi) ecosystems and wrapped-token models, mismatched contract states or unsynchronized oracle updates can result in the duplication or loss of digital assets. Moreover, inadequate validation of cross-chain events can enable attackers to trigger false deposits or withdrawals. These risks highlight the importance of formally verified contracts, consistent state synchronization, and runtime monitoring tools capable of detecting cross-chain inconsistencies before they propagate [62, 63]. Consider, for instance, a cross-chain token transfer from blockchain A to blockchain B using a lock-and-mint mechanism [39], which relies on the smart contracts shown in Figures 5, 6, and 7. Such a scenario may lead to double-spending vulnerabilities because the underlying smart contract implementations are flawed. Figure 4 illustrates a sequence of events that can give rise to these issues. Suppose Alice wants to transfer 10 tokens from blockchain A to blockchain B. To do this, she will first need to approve that the bridge can transfer her tokens calling the `approve` function of *Token* contract (line 20, Figure 5) setting the address of the bridge contract deployed in the blockchain A and the amount of token to transfer, i.e. the 10 units. Once approved, she can invoke the `lockTokens` function of the Bridge A contract (line 15, Figure 6). This call emits a Locked event to trigger the bridge logic, and then transfers Alice's tokens to the Bridge A contract, where they are locked. Subsequently, the bridge is triggered and invokes the `addUserForMint` function (line 6, Figure 7) in the Bridge B contract deployed on blockchain B that allows minting tokens for Alice and the `mintFor` function (line 11, Figure 7) to create the wrapped version of the token units in the

```
contract Token {

    mapping(address => uint256) public balanceOf;
    mapping(address => mapping(address => uint256)) public allowanceTransferMap;

    constructor(uint256 initial) {
        balanceOf[msg.sender] = initial;
    }

    function transferFrom(address from, address to, uint256 amount) public returns (bool) {
        require(balanceOf[from] >= amount, "No money, no cry!");
        require(allowanceTransferMap[from][msg.sender] >= amount, "What are u doing? It is not allowed!");
        allowanceTransferMap[from][msg.sender] -= amount;

        balanceOf[from] -= amount;
        balanceOf[to] += amount;
        return true;
    }

    function approve(address spender, uint256 amount) public returns (bool) {
        allowanceTransferMap[msg.sender][spender] = amount;
        return true;
    }

    // ... other functions and fields
}
```

:

Figure 5: Token Contract.

blockchain B.

The first issue is that the *Locked* event is emitted *before* the tokens are actually transferred, i.e. locked, to the Bridge A contract. This means that if the transfer fails, the token could still be minted on blockchain B while Alice retains her original tokens on blockchain A, effectively resulting in double tokens. The second issue is that once Alice is authorized for minting by Bridge B, no additional checks are performed. As a result, anyone can call the `mintFor` function multiple times, creating duplicate tokens for Alice and ultimately leading to double spending. The third issue is that the data exchanged between the bridge components contains no information about the specific blockchains involved in the transfer. As a result, a locking event could be redirected or reused on a third blockchain C, or on any other blockchain, enabling an additional minting process and leading to double spending.

This highlights that preventing double spending is not only a matter of blockchain protocol design but also critically depends on the correct implementation of applications and smart contracts. Even when the underlying cross-chain mechanism is sound, subtle flaws in contract logic, such as the order of events, missing checks, or improper handling of state changes, can create opportunities for token duplication and financial loss. Therefore, careful design, rigorous auditing, and comprehensive verification of smart contracts are essential to ensure that assets remain secure across multiple blockchains and that double-spending is effectively prevented.

6. Conclusion

Traditional cash systems rely on trusted intermediaries to ensure security and prevent double-spending. However, this trust can be questioned, as for example happens in the Frank Abagnale Jr.'s exploit that we proposed in Section 1. Similarly, most digital cash systems continue to operate within the same paradigm, relying heavily on traditional banking and digital financial institutions.

Blockchain technology represents a significant step forward, offering a decentralized and transparent alternative that challenges these conventional frameworks. A blockchain employs a trustless environment to ensure transaction validity and effectively mitigate the problem of double spending, without

```solidity
contract BridgeContractA {
    Token public token;
    uint256 public nonce;

    event Locked(
        address indexed user,
        uint256 amount,
        bytes32 indexed depositId
    );

    constructor(Token _token) {
        token = _token;
    }

    function lockTokens(uint256 amount) external {

        bytes32 depositId = keccak256(
            abi.encodePacked(msg.sender, amount, nonce, block.chainid)
        );

        nonce++;

        emit Locked(msg.sender, amount, depositId);

        bool ok = token.transferFrom(msg.sender, address(this), amount);
        require(ok, "Ehi, transferFrom is failed!");
    }

    // ... other functions and fields
}
```

:

Figure 6: Buggy Bridge Contract A.

```solidity
contract BridgeContractB {
    mapping(address => uint256) public wrappedBalance;
    mapping(address => bool) public allowedUsers;

    function addUserForMint(address user) external onlyAdmin {
        allowedUsers[user] = true;
    }

    event Minted(address indexed user, uint256 amount, bytes32 depositId);

    function mintFor(address user, uint256 amount, bytes32 depositId) external {
        require(allowedUsers[user], "User not allowed!");
        wrappedBalance[user] += amount;
        emit Minted(user, amount, depositId);
    }

    // ... other functions and fields
}
```

:

Figure 7: Buggy Bridge Contract B.

the need for intermediaries. However, as blockchain ecosystems grow, cross-chain interoperability poses several challenges and new double spending scenarios. Indeed, these ecosystems typically require some degree of trust because individual blockchains can ensure the integrity of inter-chain interactions on their own. Although the topic has received some attention, it is often addressed in broader contexts of *blockchain interoperability* in the current state of the art, with only a few more specific studies on prevention and possible attack vectors, and there is also a lack of studies on adequate verification tools.

Acknowledgments

Work partially supported by SERICS (PE00000014 - CUP H73C2200089001) project funded by PNRR NextGeneration EU.

Declaration on Generative AI

During the preparation of this work, the authors used AI-based tools (*Grammarly*, *ChatGPT*) in order to: Grammar and spelling check, Paraphrase and reword. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] J. Silver-Greenberg, N. D. Schwartz, Visa and MasterCard Investigate Data Breach, 2012. URL: https://www.nytimes.com/2012/03/31/business/mastercard-and-visa-look-into-possible-attack.html, The New York Times - March 20, 2012.

[2] Federal Deposit Insurance Corporation, Failed Bank List, 2025. URL: https://www.fdic.gov/bank-failures/failed-bank-list, accessed 04/2025.

[3] K. Russell, C. Zhang, 3 Failed Banks This Year Were Bigger Than 25 That Crumbled in 2008, 2023. URL: https://www.nytimes.com/interactive/2023/business/bank-failures-svb-first-republic-signature.html, The New York Times - May 1, 2023.

[4] D. Chaum, Blind signatures for untraceable payments, in: D. Chaum, R. L. Rivest, A. T. Sherman (Eds.), Advances in Cryptology, Springer US, Boston, MA, 1983, pp. 199–203.

[5] A. Back, Hashcash - A Denial of Service Counter-Measure (2002). http://www.hashcash.org/hashcash.pdf. Accessed 06/2025.

[6] W. Dai, B-money, 1998. https://nakamotoinstitute.org/library/b-money/. Accessed 06/2025.

[7] N. Szabo, Bit gold, 2005. https://nakamotoinstitute.org/library/bit-gold/. Accessed 06/2025.

[8] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, 2008. https://bitcoin.org/bitcoin.pdf Accessed: 04/2025.

[9] E. M. Uslaner, Trust and the economic crisis of 2008, Corporate Reputation Review 13 (2010) 110–123. doi:doi.org/10.1057/crr.2010.8.

[10] G. Wood, et al., Ethereum: A secure decentralised generalised transaction ledger, 2014. Ethereum project yellow paper. https://ethereum.github.io/yellowpaper/paper.pdf Accessed: 04/2025.

[11] L. Lamport, R. Shostak, M. Pease, The Byzantine Generals Problem, ACM Trans. Program. Lang. Syst. 4 (1982) 382–401. doi:10.1145/357172.357176.

[12] S. Sayeed, H. Marco-Gisbert, Assessing blockchain consensus and security mechanisms against the 51% attack, Applied sciences 9 (2019) 1788.

[13] K. Nicolas, Y. Wang, G. C. Giakos, B. Wei, H. Shen, Blockchain system defensive overview for double-spend and selfish mining attacks: A systematic approach, IEEE Access 9 (2020) 3838–3857.

[14] Y. Marcus, E. Heilman, S. Goldberg, Low-resource eclipse attacks on ethereum's peer-to-peer network, Cryptology ePrint Archive (2018).

[15] E. Deirmentzoglou, G. Papakyriakopoulos, C. Patsakis, A survey on long-range attacks for proof of stake protocols, IEEE access 7 (2019) 28712–28725.

[16] M. Saad, J. Spaulding, L. Njilla, C. Kamhoua, S. Shetty, D. Nyang, D. Mohaisen, Exploring the attack surface of blockchain: A comprehensive survey, IEEE Communications Surveys & Tutorials 22 (2020) 1977–2008.

[17] C. Pérez-Solà, S. Delgado-Segura, G. Navarro-Arribas, J. Herrera-Joancomartí, Double-spending prevention for bitcoin zero-confirmation transactions, International Journal of Information Security 18 (2019) 451–463.

[18] K. Nicolas, Y. Wang, G. C. Giakos, Comprehensive overview of selfish mining and double spending

attack countermeasures, in: 2019 IEEE 40th Sarnoff Symposium, 2019, pp. 1–6. doi:10.1109/Sarnoff47838.2019.9067821.

[19] A. Kumar, B. Kumar Sah, T. Mehrotra, G. K. Rajput, A review on double spending problem in blockchain, in: 2023 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES), 2023, pp. 881–889. doi:10.1109/CISES58720.2023.10183579.

[20] M. Iqbal, R. Matulevičius, Exploring sybil and double-spending risks in blockchain systems, IEEE Access 9 (2021) 76153–76177. doi:10.1109/ACCESS.2021.3081998.

[21] K. Nicolas, Y. Wang, G. C. Giakos, B. Wei, H. Shen, Blockchain system defensive overview for double-spend and selfish mining attacks: A systematic approach, IEEE Access 9 (2021) 3838–3857. doi:10.1109/ACCESS.2020.3047365.

[22] M. Dotan, Y.-A. Pignolet, S. Schmid, S. Tochner, A. Zohar, Sok: cryptocurrency networking context, state-of-the-art, challenges, in: Proceedings of the 15th International Conference on Availability, Reliability and Security, ARES '20, Association for Computing Machinery, New York, NY, USA, 2020. URL: https://doi.org/10.1145/3407023.3407043. doi:10.1145/3407023.3407043.

[23] L. Olivieri, L. Pasetto, L. Negrini, P. Ferrara, et al., European union data act and blockchain technology: Challenges and new directions, in: CEUR WORKSHOP PROCEEDINGS, volume 3791, CEUR-WS, 2024. URL: https://ceur-ws.org/Vol-3791/paper30.pdf.

[24] L. Olivieri, L. Pasetto, Towards compliance of smart contracts with the european union data act, in: CEUR Workshop Proceedings, volume 3629, CEUR-WS, 2024. URL: https://ceur-ws.org/Vol-3629/paper10.pdf.

[25] S. Schulte, M. Sigwart, P. Frauenthaler, M. Borkowski, Towards blockchain interoperability, in: Business Process Management: Blockchain and Central and Eastern Europe Forum: BPM 2019 Blockchain and CEE Forum, Vienna, Austria, September 1–6, 2019, Proceedings 17, Springer, 2019, pp. 3–10.

[26] H. Yuan, S. Fei, Z. Yan, Technologies of blockchain interoperability: A survey, Digital Communications and Networks 11 (2025) 210–224. doi:https://doi.org/10.1016/j.dcan.2023.07.008.

[27] T. Haugum, B. Hoff, M. Alsadi, J. Li, Security and privacy challenges in blockchain interoperability - a multivocal literature review, in: Proceedings of the 26th International Conference on Evaluation and Assessment in Software Engineering, EASE '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 347–356. doi:10.1145/3530019.3531345.

[28] K. Sai, D. Tipper, Disincentivizing double spend attacks across interoperable blockchains, in: 2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), 2019, pp. 36–45. doi:10.1109/TPS-ISA48467.2019.00014.

[29] Z. Sun, Z. Li, X. Peng, X. Luo, M. Jiang, H. Zhou, Y. Zhang, Doubleup roll: Double-spending in arbitrum by rolling it back, in: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 2577–2590. doi:10.1145/3658644.3690256.

[30] L. Olivieri, F. Spoto, Software verification challenges in the blockchain ecosystem, International Journal on Software Tools for Technology Transfer 26 (2024) 431–444. URL: https://doi.org/10.1007/s10009-024-00758-x. doi:10.1007/s10009-024-00758-x.

[31] L. Olivieri, A. Mukherjee, N. Chaki, A. Cortesi, Cross-chain smart contracts and dapps verification by static analysis: Limits and challenges, in: CEUR Workshop Proceedings, volume 3962, 2025. URL: https://ceur-ws.org/Vol-3962/paper16.pdf.

[32] Z. Sun, Z. Li, X. Peng, X. Luo, M. Jiang, H. Zhou, Y. Zhang, Doubleup roll: Double-spending in arbitrum by rolling it back, in: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, 2024, pp. 2577–2590.

[33] J. S. Notland, J. Li, M. Nowostawski, P. H. Haro, Sok: Cross-chain bridging architectural design flaws and mitigations, arXiv preprint arXiv:2403.00405 (2024).

[34] F. H. Bappy, T. Islam, K. Hasan, J. S. Park, C. Caicedo, Impact of conflicting transactions in blockchain: Detecting and mitigating potential attacks, in: GLOBECOM 2024-2024 IEEE Global Communications Conference, IEEE, 2024, pp. 4132–4137.

[35] V. Chithanuru, M. Ramaiah, An anomaly detection on blockchain infrastructure using artificial intelligence techniques: Challenges and future directions–a review, Concurrency and Computation: Practice and Experience 35 (2023) e7724.
[36] A. Mukherjee, L. Olivieri, N. Chaki, A. Cortesi, Double-spending attacks in cross-blockchain ecosystems, Blockchain: Research and Applications (2025) 100378.
[37] J. Kwon, E. Buchman, Cosmos whitepaper, A Netw. Distrib. Ledgers 27 (2019) 1–32.
[38] G. Wood, Polkadot: Vision for a heterogeneous multi-chain framework, White paper 21 (2016) 4662.
[39] L. Olivieri, A. Mukherjee, N. Chaki, A. Cortesi, Blockchain interoperability through bridges: A token transfer perspective, in: 2024 6th International Conference on Blockchain Computing and Applications (BCCA), 2024, pp. 742–748. doi:10.1109/BCCA62388.2024.10844425.
[40] S.-S. Lee, A. Murashkin, M. Derka, J. Gorzny, Sok: Not quite water under the bridge: Review of cross-chain bridge hacks, in: 2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), 2023, pp. 1–14. doi:10.1109/ICBC56567.2023.10174993.
[41] M. Herlihy, Atomic cross-chain swaps, in: Proceedings of the 2018 ACM symposium on principles of distributed computing, 2018, pp. 245–254.
[42] M. Bartoletti, R. Zunino, A theoretical basis for mev, in: C. Garman, P. Moreno-Sanchez (Eds.), Financial Cryptography and Data Security, Springer Nature Switzerland, Cham, 2026, pp. 225–242.
[43] M. Bartoletti, R. Marchesin, R. Zunino, Defi composability as mev non-interference, in: J. Clark, E. Shi (Eds.), Financial Cryptography and Data Security, Springer Nature Switzerland, Cham, 2025, pp. 369–387.
[44] S. Guesmi, C. Piazza, S. Rossi, Noninterference analysis for smart contracts: Would you bet on it? 3791 (2024). URL: https://ceur-ws.org/Vol-3791/paper12.pdf.
[45] G. Caldarelli, Understanding the blockchain oracle problem: A call for action, Information 11 (2020) 509.
[46] G. Caldarelli, C. Rossignoli, A. Zardimi, Oracle trust models for blockchain-based applications. an early standardization, in: 2023 IEEE International Conference on Technology Management, Operations and Decisions (ICTMOD), 2023, pp. 1–6. doi:10.1109/ICTMOD59086.2023.10438142.
[47] H. Al-Breiki, M. H. U. Rehman, K. Salah, D. Svetinovic, Trustworthy blockchain oracles: review, comparison, and open research challenges, IEEE access 8 (2020) 85675–85685.
[48] A. Hassan, I. Makhdoom, W. Iqbal, A. Ahmad, A. Raza, From trust to truth: Advancements in mitigating the blockchain oracle problem, Journal of Network and Computer Applications 217 (2023) 103672.
[49] A. Albizri, D. Appelbaum, Trust but verify: The oracle paradox of blockchain smart contracts, Journal of Information Systems 35 (2021) 1–16.
[50] S. K. Lo, X. Xu, M. Staples, L. Yao, Reliability analysis for blockchain oracles, Computers & electrical engineering 83 (2020) 106582.
[51] G. Caldarelli, Overview of blockchain oracle research, Future Internet 14 (2022) 175.
[52] K. Wang, Y. Li, C. Wang, J. Gao, Z. Guan, Z. Chen, Xguard: Detecting inconsistency behaviors of crosschain bridges, in: Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering, FSE 2024, Association for Computing Machinery, New York, NY, USA, 2024, p. 612–616. URL: https://doi.org/10.1145/3663529.3663809. doi:10.1145/3663529.3663809.
[53] Z. Liao, Y. Nan, H. Liang, S. Hao, J. Zhai, J. Wu, Z. Zheng, Smartaxe: Detecting cross-chain vulnerabilities in bridge smart contracts via fine-grained static analysis, Proc. ACM Softw. Eng. 1 (2024). URL: https://doi.org/10.1145/3643738. doi:10.1145/3643738.
[54] J. Zhang, J. Gao, Y. Li, Z. Chen, Z. Guan, Z. Chen, Xscope: Hunting for cross-chain bridge attacks, in: Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering, ASE '22, Association for Computing Machinery, New York, NY, USA, 2023. URL: https://doi.org/10.1145/3551349.3559520. doi:10.1145/3551349.3559520.
[55] T.-D. Tran, K. A. Vo, D. T. Phan, C. N. Tan, V.-H. Pham, Chainsniper: A machine learning approach for auditing cross-chain smart contracts, in: Proceedings of the 2024 9th International Conference

on Intelligent Information Technology, ICIIT '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 223–230. URL: https://doi.org/10.1145/3654522.3654577. doi:10.1145/3654522.3654577.

[56] A. Augusto, R. Belchior, J. Pfannschmidt, A. Vasconcelos, M. Correia, Xchainwatcher: Monitoring and identifying attacks in cross-chain bridges, arXiv preprint arXiv:2410.02029 (2024).

[57] M. Eshghie, C. Artho, H. Stammler, W. Ahrendt, T. Hildebrandt, G. Schneider, Highguard: Cross-chain business logic monitoring of smart contracts, in: Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering, ASE '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 2378–2381. URL: https://doi.org/10.1145/3691620.3695356. doi:10.1145/3691620.3695356.

[58] DefiLlama, Total Value Hacked in Bridges, 2024. https://defillama.com/hacks Accessed 06/2025.

[59] C. G. Harris, Cross-chain technologies: Challenges and opportunties for blockchain interoperability, in: 2023 IEEE International Conference on Omni-layer Intelligent Systems (COINS), 2023, pp. 1–6. doi:10.1109/COINS57856.2023.10189298.

[60] Q. Zhao, Y. Wang, B. Yang, K. Shang, M. Sun, H. Wang, Z. Yang, H. Xin, A comprehensive overview of security vulnerability penetration methods in blockchain cross-chain bridges, Authorea Preprints (2023).

[61] G. Wagner, Double spending bug in Polygon's Plasma bridge, 2021. URL: https://gerhard-wagner.medium.com/double-spending-bug-in-polygons-plasma-bridge-2e0954ccadf1, (Accessed 04/2025).

[62] Z. Liao, Y. Nan, H. Liang, S. Hao, J. Zhai, J. Wu, Z. Zheng, Smartaxe: Detecting cross-chain vulnerabilities in bridge smart contracts via fine-grained static analysis, in: Proceedings of the 2024 ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE), 2024.

[63] M. Eshghie, C. Artho, H. Stammler, W. Ahrendt, T. Hildebrandt, G. Schneider, Highguard: Cross-chain business logic monitoring of smart contracts, in: Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering (ASE), 2024.