SoK: Security and Privacy of Blockchain Interoperability: Extended Version

André Augusto *, Rafael Belchior *, Miguel Correia †,
André Vasconcelos †, Luyao Zhang † Thomas Hardjono †
Instituto Superior Técnico *INESC-ID*
MIT Connection Science *Duke Kunshan University*

***Abstract*—Recent years have witnessed significant advancements in cross-chain technology. However, the field faces two pressing challenges. On the one hand, hacks on cross-chain bridges have led to monetary losses of around 3.1 billion USD, highlighting flaws in security models governing interoperability mechanisms and the ineffectiveness of incident response frameworks. On the other hand, users and bridge operators experience restricted privacy, which broadens the potential attack surface.**

In this paper, we present the most comprehensive study to date on the security and privacy of blockchain interoperability. We employ a systematic literature review, yielding a corpus of 212 relevant documents, including 58 academic papers and 154 gray literature documents, out of a pool of 531 results. We systematically categorize 57 interoperability solutions based on a novel security and privacy taxonomy. Our dataset, comprising academic research, disclosures from bug bounty programs, and audit reports, exposes 45 cross-chain vulnerabilities, 4 privacy leaks, and 92 mitigation strategies. Leveraging this data, we analyze 18 notable bridge hacks accounting for over 2.9 billion USD in losses, mapping them to the identified vulnerabilities.

Our findings reveal that a substantial portion (65.8%) of stolen funds originates from projects secured by intermediary permissioned networks with unsecured cryptographic key operations. Privacy-wise, we demonstrate that achieving unlinkability in cross-chain transactions is contingent on the underlying ledgers providing some form of confidentiality. Our study offers 17 critical insights into the security and privacy of cross-chain systems. We pinpoint promising future research directions, underscoring the urgency of enhancing security and privacy efforts in cross-chain technology. The identified improvements have the potential to mitigate the financial risks associated with bridge hacks, fostering user trust in the blockchain ecosystem and, consequently, wider adoption.

Keywords: Cross-chain interoperability, Security, Privacy, Vulnerabilities, Blockchain Technology, Cryptocurrency, Financial Losses, Systematic Literature Review, Taxonomy, Mitigations.

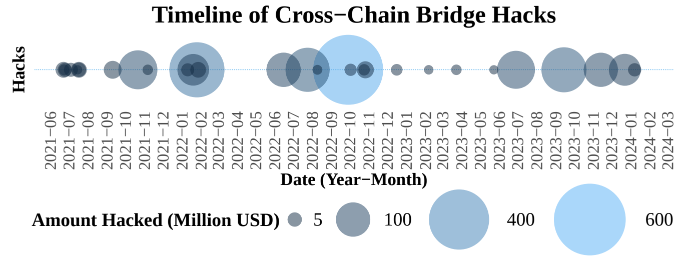


Figure 1. Timeline of cross-chain bridge hacks from May 2021 to June 2023. The dataset encompasses 33 bridge hacks and is summarized in Table J. The total amount of loss accounts for more than USD 3.2B.

1. Introduction

Blockchain interoperability is a key component for realizing the full potential of blockchain technology. As the landscape evolves, interoperability is gaining momentum in use cases including bridging liquidity fragmentation, optimizing decentralized exchanges (DEX) trades, enhancing scalability through mechanisms like sharding [1], extending through sidechains [2], and enabling asset exchanges and transfers across platforms [3]. Taking a step back to 1996, Wegner postulated [4]: "interoperability is the ability of two or more software components to cooperate despite differences in language, interface, and execution platforms". However, achieving interoperability across blockchains – distributed systems where mutual trust is often absent – adds a dimension of complexity. Here, the challenge is not merely syncing *n* software components but rather integrating *n* distributed systems, each with its unique challenges encompassing safety, liveness, accountability, and centralization [5], [6]. Such orchestration is realized using interoperability mechanisms (IMs). The differing transactional models, consensus mechanisms, and cryptographic primitives across networks only escalate this challenge. Despite these hurdles, the domain has seen prolific contributions from scholars, providing solutions, novel architectures, and varied use cases [7]–[15]. A recurring theme from these

---

∗. The first two authors contributed equally to this work. Rafael conducted this research while at MIT Connection Science supported by a Fulbright Scholarship.

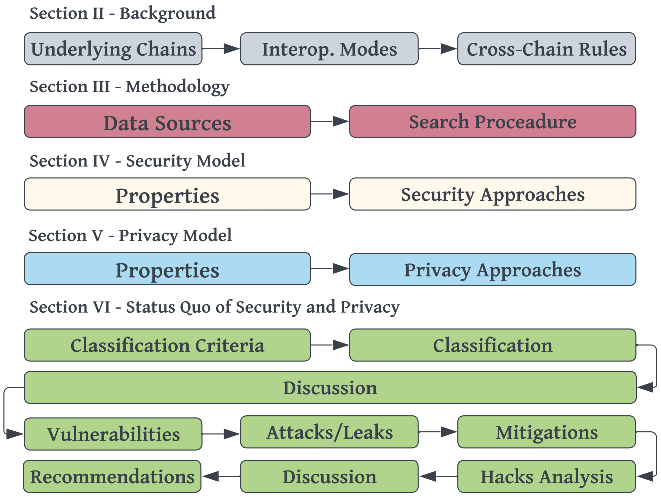


Figure 2. Section 2 presents the background knowledge necessary for understanding this paper. Section 3 shows the methodology followed to systematically analyse existing work. We present our security and privacy models in Sections 4 and 5, respectively. These include relevant properties, security- and privacy-enabler solutions, and the analysis of vulnerabilities, attacks, and leaks on cross-chain systems. The classification of the selected corpus of studies is presented in Section 6 along with a discussion of the results.

studies underscores the pressing need for rigorous research on security and privacy in IMs.

1.1. Motivation

The importance of studying security in interoperability cannot be understated. Since May 2021, the mounting losses due to bridge hacks have exceeded USD 3B, as illustrated in Figure 1. According to Immunefi [16], white-hat hackers have been compensated over USD 20M through bug bounty programs, preventing potential losses of a staggering USD 1B. Moreover, cross-chain bridge hacks have catapulted to the top of the DeFi incidents leaderboard [17]–[19], emerging as the preferred target of cybercriminals. The present scenario, as of mid-2023, paints a grim picture with rampant hacks [20]–[23]. Consequently, the total value locked (TVL) in cross-chain bridges has nose-dived from its zenith at USD 58B in early 2022, to a mere USD 4.5B by October 2023, a downfall also attributed to diminishing asset prices in the bearish market [24], [25]. We hypothesize that the intertwined cross-chain systems, in conjunction with the already well-studied vulnerability-prone smart contracts, be it at the bytecode or higher-level language dimensions [26], have amplified the risk exposure of these protocols. The fact that those protocols are attractive honeypots makes them keenly pursued targets, across the three interoperability modes we examine: asset exchanges, asset transfers, and data transfers [7], [27]. If this issue is not comprehensively tackled, the trajectory suggests a future with no bridge left uncompromised. Furthermore, malevolent entities use cross-chain on and off ramps to launder stolen assets—sometimes from other bridges [28], attempting to circumvent deny lists and complicating forensic investigations, given several forensic tools available widespread [29], [30]. Even though most of the funds stolen in interoperability solutions come from cross-chain bridges, other interoperability schemes also originate numerous vulnerabilities [31]–[33].

In this paper, we capture the security and privacy approaches of all solutions and establish a comprehensive understanding of cross-chain security and privacy—a subject currently scattered throughout various sources in the literature. Strikingly, similar vulnerabilities can often denote distinct notions of security and privacy. As such, our approach leans heavily on a methodological survey, where we gather, evaluate, and study academic papers and grey literature, spanning blog posts, whitepapers, and audit reports. To our understanding, our effort represents the most extensive survey of blockchain interoperability to date. We distil insights from the academic discourse, extant cross-chain attacks—from a theoretical and practical perspective, and their subsequent implications. Additionally, we propose a set of mitigation strategies and best practices tailored to the corpus of vulnerabilities we have studied.

1.2. Research Questions

This paper answers several research questions:

RQ1 – What are the different security- and privacy-centric goals used in blockchain interoperability, and what are the technical building blocks that guarantee them?.

Securely interoperating different blockchains is a challenging task. It involves establishing a new security boundary that depends on the security of at least two existing networks and involves multiple design trade-offs [11], [27], [33], [34]. Similarly, considering a set of blockchains with different privacy guarantees, it is unclear what privacy is in the context of multiple systems. Also supported by a recent work [35], we reckon that there is too much privacy for criminals and too little privacy for general users – i.e., a non-accountable ecosystem cannot penalize malicious actors and thus renders the system unfair.

Security-wise, we propose a set of properties from the distributed system literature to classify interoperability solutions based on the specific security approaches employed and how they guarantee safety and liveness in each protocol. The most straightforward safety violation is the absence of atomicity in cross-chain transactions, which can lead to "double-spending". Some cross-chain deal [36] and atomic swap protocols [37]–[41], instead of relying on the all-or-nothing property, state that everything is fine if honest nodes do not end up worse off than how they started the protocol.

In terms of privacy, for example, Hash-Time Lock Contracts (HTLCs) [38] suffer from transaction linkability problems. Transactions in different chains can disclose cross-chain interactions between parties through a value published on both chains [42]. In asset transfer bridges, the amount locked in one chain can be linked to the amount minted on the destination chain, leading to the same problem. Additionally, the lack of privacy for bridge operators increases the attack surface to those entities [43].

In this paper, we aim to explore the relevant properties and approaches to guarantee the different levels of security and privacy in blockchain interoperability.

RQ2 – What are the cross-chain vulnerabilities, attack vectors, privacy leaks, and mitigations currently known, and how are they mapped to past incidents?.

Our second research inquiry compels us to investigate cross-chain attacks, focusing on the vulnerabilities that give rise to them. We categorize these identified vulnerabilities into four distinct security layers explained in Section 4.2. In addition to the theoretical research, we examine real-world cross-chain hacks, which collectively account for over 3 billion USD, and compare them with academic studies. We pinpoint the disparities between existing research findings and their practical application and map each vulnerability to possible mitigations.

RQ3 – Based on the existing gaps, what are potential best practices and avenues for future research to enhance the security and privacy of cross-chain protocols?.

When it comes to practical considerations, when combining an extreme privacy- or security-focused blockchain with a lesser one, the resultant degree of security or privacy is likely to be minimal. Additionally, in a world where achieving a balance between transparency and privacy is imperative, and total privacy may hinder the ability to trace transactions, potentially affecting investigations into security breaches, the ideal level of privacy for cross-chain scenarios remains unclear. In a time when the industry is actively seeking stability, we observe that the design of cross-chain solutions remains largely ad hoc, with each solution custom-crafted for specific blockchains or applications. Through a comprehensive analysis of existing studies, we put forth a collection of best practices and future research avenues. In this paper, we will delve into these intricate questions and furnish initial insights that protocol designers, developers, and analysts can use as a foundation for further research and development.

1.3. Contributions

This paper provides the following contributions:

- **Systematization of knowledge.** Systematizes properties and approaches of secure and privacy-enhancing cross-chain protocols by curating relevant academic literature and real-world data, including cross-chain hacks and other audit reports, and provides an in-depth analysis.
- **Academia-industry synergy.** Bridge the divides between academic research and industry application, correlating theoretical vulnerabilities with past real-world incidents.
- **Strategic insights.** Identifies lessons learned, highlights emerging research directions, and proposes a set of mitigations and best practices, enabling practitioners to build secure and private cross-chain bridges.

This manuscript systematically consolidates and builds upon existing literature, illuminating novel insights and advancements within the research domain. The organizational framework of this document is depicted in Figure 2. Section 2 provides a primer on blockchain interoperability. Section 3 introduces a rigorous methodology, systematically adopted to constructively assess and incorporate prior research. Section 4 and Section 5 detail our advanced security and privacy models, respectively. These sections not only spotlight pivotal properties but also innovative mechanisms for enhancing security and privacy, along with proactive strategies to optimize potential challenges in cross-chain systems. Section 6 offers a structured categorization and an insightful analysis of the curated body of interoperability research. Section 7 briefly discusses the contribution to related literature. Concluding this discourse, Section 8 presents our forward-looking remarks and insights.

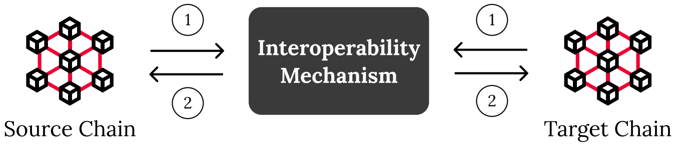


Figure 3. Blockchain interoperability studies the flow of data and assets between two networks powered by an Interoperability Mechanism (IM). The actor(s) that take the role of IM depend on the Security Approach of the solution (cf. Section 4). Legend: ① Fetch data and ② Issue Transactions.

Data and Code Availability: The data and code for ensuring replicability are available on GitHub, accessible at the following URL: https://github.com/RafaelAPB/SoKSPBlockchainInterop.

2. A Primer on Blockchain Interoperability

This section presents a succinct introduction to blockchain interoperability, which is necessary for understanding the rest of this paper. Blockchain interoperability allows data and value to be sent across a set of different domains. Besides distributed ledgers, these domains can have the form of centralized databases, mainstream systems, or any other distributed system. This paper focuses mainly on cross-chain systems where domains are distributed ledgers $\{l_1, l_2, ..., l_k\} \in \mathcal{L}$.

2.1. The Source of Truth - Underlying Blockchains

In this paper, while the primary emphasis is not on the security of the underlying networks of cross-chain protocols, it is imperative to recognize their pivotal role as critical dependencies for these protocols. Consequently, we provide a succinct and scientific overview elucidating the relevance of the network and consensus layers in the context of interoperability studies.

The finality of the source chain in an interoperability solution is critical: chains with Nakamoto-based consensus (i.e., a probabilistic finality algorithm called proof of work – PoW) are subject to forks. Proof of Stake-based (PoS) chains are sensitive to long-range attacks [44]. As a common vulnerability, forks can be created in these protocols, as more valid blocks are mined in parallel suffixing a determined block in the chain. If these block headers are relayed to the target chain before being considered final, actions based on them should not be successful in guaranteeing the absence of safety violations – e.g., double spending or other violations of cross-chain logic [32]. This main chain identification increases the complexity of the bridge contract in the destination chain. On the other hand, chains with instant or near-instant finality such as PBFT-based [45], do not suffer from the same problem. Blocks are only added to the blockchain when they are already considered final. However, the verification cost and complexity differ as one needs to know the validation committee at each point to validate the corresponding attestations - this does not work for dynamic committees that most blockchains use.

A possible attack is a cross-chain 51% attack, where the attacker creates valid block headers faster than the rest of the network, and exploits the difference in state between before and after the attack. For example, the attacker sends funds to a bridge (spends the funds on the source blockchain), and sends a Merkle proof and block header to the relay contract. After that, he conducts a 51% attack and gets the funds back on the source chain. However, the bridge does not revert, yielding a cross-chain double spend. To the best of our knowledge, this specific attack has not been verified in practice.

2.2. Interoperability Modes

The literature [1], [46]–[49] agrees on the three existing interoperability modes: asset exchanges (AE), data transfers (DT), and asset transfers (AT). Different interoperation modes require different protocol architectures, and consequently different security and privacy guarantees.

Consider accounts $\mathcal{A}_1$ and $\mathcal{A}_2$ in domains l_1 and l_2 (typically domains are ledgers, but note that for arbitrary cross-chain interoperability, domains may be centralized systems). Asset exchange protocols allow untrusted parties to atomically exchange assets. For example, asset X owned by $\mathcal{A}_1$ on ledger l_1, can be exchanged for asset Y owned by $\mathcal{A}_2$ on ledger l_2. This is achieved by issuing local transactions in both blockchains with the assistance of hash-locks (the proof p) [10]. An asset exchange can be mediated by a trusted party, or run directly between both parties through an off-chain communication channel.

Asset transfer protocols encompass locking or burning an asset in the source chain and creating (minting) a representation of that asset in the target chain – we call it the *lock-mint* or *burn-mint* pattern, respectively. In practice, the process of locking is transferring the asset to an escrow controlled by a smart contract, a centralized entity, or a set of parties through a multi-signature. Once the asset is locked in the source chain, the verification occurs in the target chain. The verification can be done by replicating the source chain's consensus mechanism in the target chain [50], [51] or using a proof-based mechanism such as zero-knowledge proofs [44], [52], [53]. An alternative mechanism is leveraging liquidity pools on both chains [7], [54], where no asset is minted, but rather several native assets are unlocked and sent to the user. However, these escrows create honeypots that incentivize attackers to break through them.

Data Transfers generalize interoperability. Information written in one domain can be transferred or copied to another (typically accompanied by a proof, for example, the payload of a blockchain view [55]). Usually, distributed ledger technology (DLT) Gateways are used to facilitate this process, running a gateway-to-gateway protocol [56]. Different interoperability modes are different classes of cross-chain rules.

2.3. Cross-Chain Events, Transactions and Rules

The concepts of cross-chain events, transactions, and rules are important to understand this work. Transactions issued in one domain trigger internal state changes and emit events based on the operations performed. Cross-chain events are composed of native and non-native domain attributes. Native attributes are retrieved from the events emitted in the underlying domains. Non-native attributes are additional metadata that only hold relevance in cross-chain environments, such as a domain identifier, a global clock, a token price, or other off-chain information. Metadata is published on-chain by decentralized oracles. Its correctness is measured by the correctness of the oracle network and according to the agreement between entities to perform *cctxs*.

Definition 1 (Valid Cross-Chain Event)**.** *A cross-chain event e is valid iff its metadata is correct***, and every local transaction $t \in e$ is final.*

The composition of multiple cross-chain events stands for state changes across several domains. We call this composition a cross-chain transaction (*cctx*). To evaluate the validity of a *cctx*, events must be verified against *cross-chain rules* that define the expected behavior. A rule for an asset transfer protocol might indicate that there must not be an event minting an asset in l_2 before an event locking the corresponding asset in l_1. Given some business logic, one can create arbitrarily complex cross-chain rules. We refer the reader for a formal treatment [32] with a specific example of cross-chain rules on [57], and to the Appendix A.

Definition 2 (Valid Cross-Chain Transaction)**.** *A cross-chain transaction cctxis valid iff every cross-chain event $e \in cctx$ is valid, and all cross-chain events enforce the defined cross-chain rules.*

3. Research Methodology

This section presents the methodology followed to answer the research questions enumerated in Section 1. Firstly, we perform a literature review that focuses on the search for papers about security and/or privacy, including attacks,

---

$*$. if metadata can be evaluated – e.g., the price of the token being transferred is within an agreed interval incidents, or vulnerabilities in blockchain interoperability solutions. Finally, we actively search for resources in grey literature to retrieve data about recent cross-chain hacks, and incident or audit reports. The methodology is depicted in Figure 8, in Appendix B.

3.1. Data Sources

We used Google Scholar as our primary source of data given that it indexes most major digital libraries and proceedings (e.g., ACM Digital Library, IEEE Xplore, Springer Nature Lecture Notes in Computer Science), grey literature libraries (e.g., arXiv, Cryptology ePrint Archive), and other resources (e.g., books, thesis, or other online repositories).

3.2. Search Procedure

We conducted a systematic literature review by crawling papers using Google Scholar's keyword search. The search was limited to papers since 2015 due to the limited amount of research available before that period [10]. The following search query was used to search for papers within our research scope:

```
("blockchain interoperability" OR "cross-chain") AND
(“attack" OR “incident” OR “hack” OR "leaks") AND (
(“security” AND (“vulnerability” OR "mitigation"))
OR "privacy")
```

This search yielded 2010 results. We stopped searching on the 300th reference, finding no relevant papers beyond the 250th. In this first analysis, we filtered the results according to being written in English and to their title and abstract. Our final selection criteria included papers addressing blockchain interoperability security, privacy, attacks, vulnerabilities, leaks, or corresponding mitigations. We ended up with 58 studies. In addition to the retrieved results, we utilized the *snowballing* and *forward reference* techniques. We also set up Google alerts to stay informed about new papers related to "*blockchain interoperability*" and "*cross-chain*". These were retrieved manually according to the same criteria. Through this approach, we identified an additional 49 studies, which we believe cover the majority of relevant research for our study. Due to the unstructured practices in the area, we included multiple gray literature resources focusing on past cross-chain hacks, audit reports, vulnerabilities, and disclosures through bug bounty programs. Therefore, we analyzed an additional 154 relevant documents.

3.3. Practical screening criteria

In our practical screening criteria, we place paramount importance on cross-chain privacy and security, delving into the nuances of cross-chain hacks and vulnerabilities. Additionally, we give comprehensive attention to L2 solutions, particularly rollups, based on native cross-chain bridges. Conversely, our screening criteria exclude applications based on cross-chain solutions, security and privacy of individual blockchains, L2 solutions such as payment channels that lack cross-chain relevance, and security issues exclusive to smart contracts without broader cross-chain implications.

3.4. Limitations

Naturally, our methodology comes with certain limitations. In our quest for grey literature resources primarily centered on cross-chain hacks, to mitigate the potential for unsoundness or bias, we meticulously compile data from a variety of sources. In these cases, each resource undergoes thorough examination and assessment by at least two of the paper's authors to guarantee the integrity of the information. Additionally, we acknowledge that our analysis of industry solutions and associated vulnerabilities may not encompass the entirety of the landscape, primarily due to limitations in the available documentation. Nevertheless, we make diligent efforts to compile all accessible information concerning projects that collectively represent over 75% of the Total Value Locked (TVL) in cross-chain solutions[25]. Nevertheless, despite these limitations, this paper is the most extensive and comprehensive research paper conducted so far on the security and privacy of blockchain interoperability.

4. Security Model for Cross-Chain Systems

In this section, we present the properties that define a secure interoperability system while considering the requirements of the various stakeholders. Additionally, we showcase and discuss the literature according to the existing security approaches. Finally, we present the most extensive list, so far presented, of theoretical cross-chain vulnerabilities, attacks, and mitigations and map them to real-world hacks that account for more than 3 billion USD. We gather all relevant insights and propose guidelines for building secure and robust cross-chain systems.

4.1. Motivation

A common assumption for interoperability systems is that the underlying chains are trusted. The reasons are clear: if a transaction t_2 is issued on l_2 based on a rewritten transaction t_1 on l_1, there is a safety violation. For instance, consider a transaction locking an asset in l_1 and a representation minted in l_2. If t_1 reverts, the asset in l_2 becomes unbacked [58]. Figure 4 illustrates this scenario. ① A dishonest miner, which also operates a relayer, is a selfish miner [59]; ② the selfish miner forges a valid syntactic block with transactions that did not occur (e.g., locking random assets); ③ the selfish miner sends those block headers via the relayer to the source chain light client in the target chain. Consensus-wise, block $i{+}1b$ is valid despite not being included in the canonical chain, signifying the creation of a fork. The target chain should have a fork resolution mechanism to decide which block to accept [44], [50], and therefore, which transaction inclusion requests to accept.

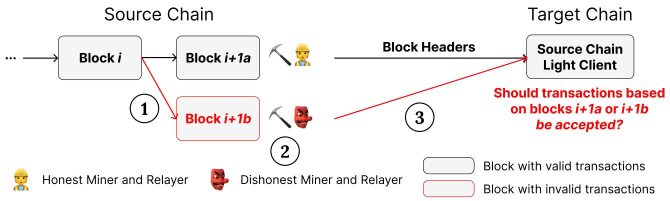

Source Chain
Target Chain
...
Block i
Block i+1a
Block i+1b
Block Headers
Source Chain Light Client
Should transactions based on blocks i+1a or i+1b be accepted?
①
②
③
Honest Miner and Relayer
Dishonest Miner and Relayer
Block with valid transactions
Block with invalid transactions

Figure 4. The underlying chain's security influences the security of the interoperability solution. Additionally, there must be effective fork resolution mechanisms to cope with less secure source chains.

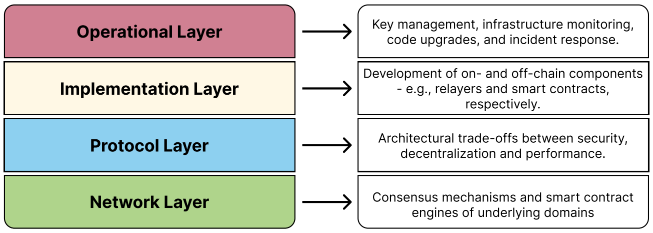

Operational Layer
Key management, infrastructure monitoring, code upgrades, and incident response.
Implementation Layer
Development of on- and off-chain components - e.g., relayers and smart contracts, respectively.
Protocol Layer
Architectural trade-offs between security, decentralization and performance.
Network Layer
Consensus mechanisms and smart contract engines of underlying domains

Figure 5. Security domains relevant for cross-chain solutions.

This theoretical attack is one of the arguments for a multichain ecosystem (i.e., blockchain engines [10]) instead of cross-chain bridges. In those systems, the execution and settlement layers are one, which results in the cross-chain state (e.g., source chain's block header) being validated by a shared stratum between instances of a blockchain engine. However, bridges between blockchain engines and between blockchain engines and external blockchains still have to be developed (e.g., Toposware [60] and Cosmos [61], or Polkadot [62] and Ethereum [63]). Multichain ecosystems will probably need cross-chain interoperability, and the security of such interactions does not seem fundamentally different from cross-chain interoperability between L1s [7].

4.2. Security Layers

The security of a cross-chain system can be decomposed into the security of several layers, as depicted in Figure 5. Existing literature supports similar breakdowns [64]. The ***Network Layer*** ● forms the bedrock. It concerns about the systems or networks that underlie a cross-chain solution. These can be distributed ledgers or even centralized databases. For instance, the chosen consensus mechanism and smart contract engines drive the security of this layer. Above that, the ***Protocol Layer*** ● addresses the different architectural decisions to build a cross-chain protocol. It includes defining the actors, their roles and responsibilities, and how the relevant security and performance properties are guaranteed. Further up the stack, we encounter the ***Implementation Layer*** ○. It encompasses the entire implementation lifecycle, including off-chain (e.g., relayers, oracles, incident response systems) and on-chain code (e.g., smart contracts, protocols) to serve as mechanisms to facilitate interoperability and on-chain contracts that execute the respective business logic. Finally, at the top, once cross-chain solutions are designed and implemented, one must ensure that it is operational and upgradeable. As in every software, off-chain and on-chain programs may have vulnerabilities that compromise their functionality. At the ***Operational Layer*** ● , specifies the procedures for deploying, maintaining, and upgrading on-chain and off-chain components. It concerns with who manages the protocol, how to monitor the infrastructure, how to update the code, and how the system reacts to external or internal unexpected events.

This paper explores vulnerabilities, attacks, and mitigations at the last three layers – i.e., as said in Section X, we assume the underlying networks are safe and live and are concerned about security at the protocol, implementation, and operational levels.

4.3. Security Properties

Based on our comprehensive literature review, we strive to propose a set of fundamental properties that characterize a secure cross-chain system, based on well-known work in the dependable computing area [65], based on fundamental cross-chain concepts [32]. The core idea behind security properties is that they depend on the underlying cross-chain logic (a set of cross-chain rules). We define three security properties for IMs. The integrity of a cross-chain system is evaluated as a function of how reliable the integrity of the data or assets is managed:

Definition 3 (Integrity)**.** *Consider an IM and a set of cross-chain rules ζ. Integrity is guaranteed iff every generated cctx respects ζ.*

Integrity depends on the use case and therefore the interoperability mode. A rule of thumb for integrity in asset transfers and asset exchanges is that double spend should not occur. Conversely, for data transfers, integrity is normally assured by enforcing every transferred data point d to have equivalent representations using a standardized blockchain view [55]. IMs must be held accountable for their actions, namely if they provide integrity. This property encompasses two facets: identification and punishment of actors, for example through slashing mechanisms [44]. Furthermore, a cross-chain protocol should ensure non-repudiation of actions regarding its participants.

Definition 4 (Accountability)**.** *An IM is accountable iff these conditions hold: 1) the metadata of any event $e \in$ cctx is public, or at least verifiable*[†]*; 2) for every integrity violation attempt in ζ, there is a mechanism to prove it; and 3) there is a third-party that can enforce punishments for proved attempts (e.g., third-party, blockchain smart contract).*

Finally, we require the availability property for IMs to be considered secure and guarantee the availability of resources and services for users and operators. Protocols must be resilient to failures and handle unexpected behavior quickly.

---

†. for example, private data verifiable through proving systems

Definition 5 (Availability)**.** *An IM guarantees availability iff it is always able to process (validate, issue, or relay) valid cctxs.*

The availability of IMs therefore depends again on the specification of cross-chain rules and, furthermore, the liveness of the underlying infrastructure.

4.4. Security Approaches

We discuss the proposed cross-chain security taxonomy based on the approaches to guarantee the properties above.

4.4.1. Trusted Third Parties. Trusted third parties can facilitate interoperability by managing the cross-chain flow between chains. Trust can be put on the reputation of the managing party [56], [57], [71], [78], or on secure enclaves with attestable computation [76]–[78].

Centralization. Centralized trusted parties can hold users' funds and issue transactions in $\mathcal{L}$ [107], or function solely as relay services [48], [56] that do not hold funds but are legally accountable for the information relayed. In rollups, centralized operators are responsible for ordering and batching transactions in the L2 and submitting them to the L1. To avoid liveness compromises, they usually allow submitting proofs directly to the L1 contract [108]. Project maintainers operate these nodes, which raises concerns about centralization, censorship and MEV. Centralization offers increased performance since no agreement between parties is needed and effectively eliminates threats from decentralized architectures [66]. Additionally, it enforces accountability as entities must comply with KYC [70]. However, *cctxs* can be censored due to bribery, coercion collusion [75]. Note that multiple centralized exchanges have been the target of cyber-attacks resulting in users experiencing fund theft (e.g., FTX [109], BXH [110], or more recently CoinEx [111]). These are due, for example, to wallet security breaches [112], DDoS attacks [113], or due to rug pulls [21].

Trusted Computation. TEEs (trusted execution environments) [114] ensure the integrity and authenticity of computation conducted within a secure enclave at the hardware level. Computation validity can be verified through local or remote attestation by external parties. TEE-based cross-chain solutions focus on protecting private keys [43], operations on key shares between operators [115] or generate proofs [43], [116], [117]. In centralized TEE-based systems, the power is solely given to the administrators [79], which can promote unfairness by censoring or colluding to maximize financial gains. A possible solution is having multiple TEE-based hosts controlled by different entities (and from different hardware providers [104]) to remove the possibility of collusion [77]. LayerZero-based applications [118], that rely on a single oracle and relayer, can also be subject to collusion attacks as shown in [119]. TEEs have, nonetheless, inherited limitations. The enclave attestation keys are provided by the manufacturer who can embed malicious code into the hardware or spoof data [79]. TEEs are also subject to other attacks discussed in previous work [120].

4.4.2. Distributed Trust. Centralized solutions are simple to implement, faster, and cheaper but incur higher security risks. A solution for this over-trusted mechanism is to rely on distributing power among multiple entities to validate *cctxs* – i.e., through an intermediary distributed network. Intermediary networks verify and maintain proof of actions of other chains [36], [80], [121], [122]. More decentralized solutions leverage consensus mechanisms such as proof of work or proof of stake, in which anyone can be a part of the voting. On the other hand, less decentralized solutions use consensus mechanisms, such as proof of authority or PBFT, where involved parties are whitelisted. In the case of small networks, Threshold Signature Schemes (TSS) and Multi-Signatures (MS) are possible solutions. There are cross-chain applications based on intermediary networks for all interoperation modes (asset exchanges [36], [37], [80], data transfers [73], [76], [85], [86], [123], and asset transfers [43], [82]–[84], [88], [89]).

Permissionless Networks. When applied to cross-chain bridges, they produce new state roots containing *ccevents* of all chains, which are sent to the destination chain, which validates requests against them [82]. Cross-chain protocols based on this architecture have to deal with increased latency and cost of execution. This is due to requiring an agreement between untrusted parties, and the issuance of transactions that pay (usually high) gas fees.

Relying on external networks to validate the state of *cctxs* makes the cross-chain protocol reliant on its security since the miners of this network act as the validators for the cross-chain protocol. In particular, when dealing with public networks, its security depends on an appropriate incentive mechanism for users who follow the protocol and slashing for malicious actors [37]. Actors should not profit more from deviating than from following the protocol. Furthermore, in case of misbehaviour, they should be held accountable. Interoperability solutions based on proof of stake networks should guarantee that the overall value protected by the protocol does not surpass the stake held by the majority of validators otherwise its cryptoeconomic security might not be enough [44], [124]. Axelar [121] adds another security layer using quadratic voting (when validating *cctxs*) to avoid vulnerabilities associated with the increasing voting power of high-stake validators [125]. Note that interoperability based on intermediary networks using custom tokens with low value is not advisable due to token price fluctuations (cf. Section 6.2).

Finally, to tackle the ad-hoc bridge design problem, Blockchain Engines were proposed [10]. This model relies on a relay chain that connects to several other smaller chains, and provides shared security and composability in that interconnected environment. Each project has a custom messaging mechanism (e.g., IBC or XCMP) that allows arbitrary communication between networks within the same ecosystem. Even though this standardizes cross-chain communication within each ecosystem, it is still required

TABLE 1. Two Tier Classification of Security Approaches in Blockchain Interoperability Academic Studies. We present the primary security approach of solutions that employ various.

Security Approach (Tier 1)	Security Approach (Tier 2)	IM Role	References	# (and %)
SA_1 Trusted Third Parties	SA_{11} Centralization	Centralized Services	[48], [66]–[76]	12 (24%)
	SA_{12} Trusted Computation	Trusted Execution Environment	[77]–[79]	3 (06%)
SA_2 Distributed Trust	SA_{21} Permissionless Network	Public Network Validators	[80]–[82]	3 (06%)
	SA_{22} Permissioned Network	Whitelisted Network Validators	[37], [43], [83]–[89]	9 (18%)
SA_3 Native State Verification	SA_{31} Inclusion Proofs	Relayers	[50], [58], [90]–[92]	5 (10%)
	SA_{32} Validity Proofs	Relayers	[52], [53], [93], [94]	4 (08%)
	SA_{33} Fraud Proofs	Relayers	None in academia	0 (00%)
SA_4 Local Verification	SA_{41} Secret- & Time-based Locks	Off-chain Communication Channel	[38], [40], [42], [95]–[106]	15 (29%)

Note: The table categorizes various security approaches (SAs) prevalent in blockchain interoperability research into two tiers. The first tier provides an overarching classification, while the second tier offers a finer granularity. The "IM Role" column denotes the component that takes the role of the Interoperability Mechanism (IM), and the "References" column cites specific studies or implementations that employ the particular approach. The final column quantifies the number and approximate percentage of papers adopting each method, visually represented using cell shading.

to address inter-ecosystem interoperability [60]–[62], [115]. An advantage of the Blockchain Engine approach is that it is easy to plug in new chains, eventually requiring the development of compatibility modules that allow communication. The main difference to the other ad-hoc approaches is this layer of shared security on top of the coordinating chain. Additionally, the relay chain can be tailor-built to specific use cases in which business logic validations occur at that level [126].

Permissioned Networks. Instead of relying on a network in which anyone can join, one can opt for a more controlled environment. In PoA [123] and TSS-based [43], [88], [127] networks, validators are whitelisted and usually controlled by reputable or trusted entities. The only requirement is the existence of an identification service where parties register beforehand [96]. However, we still highlight the necessity of securing permissioned networks using economic incentives. Firstly, the due diligence to select parties is seldom known. Secondly, reputable entities can engage in transaction censorship. Finally, they have been subject to hacks or bankruptcy (e.g., FTX, Terra, Binance). The Uniswap Assessment report [124] also raises questions regarding Wormhole's PoA network, mainly on how they guarantee expected validators' performance, protocol involvement, and SLA (service level agreements) compliance.

Nonetheless, this controlled environment simplifies paramount challenges that are still unaddressed. Relying on fewer entities decreases the latency required for any state change, be it a normal *cctx*, an upgrade, or an emergency pause transaction. This contrasts with the approach of permissionless networks, where these actions would require a long voting period enforced by the underlying governance protocol. There is a tension between the level of decentralization of a protocol and the efficiency of incident response protocols. Permissionless networks of validators can authorize transactions based on a threshold of valid signatures collected on- or off-chain and submitted in batch to trigger blockchain state changes [127].

Asset exchange protocols can leverage these schemes to build trust directly between users and operators [74], [75], [105]. The main advantage of this scheme is that no user can move funds without the agreement of the trusted validator, nor the other way around. Nevertheless, funds can be locked for a long time if the validator becomes unavailable or, in the worst-case scenario, gets compromised. A possible solution is the involvement of multiple entities to guarantee liveness even in the event of crashes. The core assumption in these protocols is that the validator is trusted – i.e., it is unclear how one could recover if each validator misbehaves. A possible solution to guarantee liveness is employing some rollback mechanism triggered either when a period elapses or by presenting proof to a trusted component through cryptographic mechanisms. These can be implemented as a timelock or a centralized disputer, respectively.

The usage of MPC to build trust in environments with mutually untrusted entities has also been proposed by [43], [103], [104], [127]. However, once trust is set, protocols require other mechanisms to validate *cctxs*, such as TSS or MS. There are also Blockchain Engines approaches based on permissioned networks. The main idea is to incorporate additional security measures at the relay chain, such as access control or data verifications in *cctxs* [76], [84]. Lastly, [83] considers a different architecture for a permissioned blockchain system, which is organized hierarchically with multiple tiers of networks that publish state in the main relay chain. The authors argue that it improves the scalability via parallel transaction execution, similar to sharding. We raise concerns about the number of hops a *cctx* would require between these tiers. On the other hand, HyperService [85] proposes a system that offers interoperability and programmability across blockchains through a blockchain of blockchains designed to provide a unified view of dApps' execution status – i.e., it is possible to write smart contracts that execute logic in multiple subnetworks atomically but with limited business logic.

4.4.3. Native State Validation. Previous approaches rely on external validation techniques, either performed by one or multiple parties, in more centralized or decentralized settings. We now dive further into security approaches that rely on proof validation within the interoperating networks.

Inclusion Proofs. We now delve into security approaches that rely on on-chain proof validation. Such schemes use proofs to provide verifiable evidence of emitted events in one chain. Relayers send block headers from the source chain to the target one. These headers serve to validate proofs provided by users and can also be verified – for instance, using zero-knowledge proofs – a prominent solution is Harmonia [44]. Users can submit requests once block headers are accepted and marked as final in the light client implementation. Inclusion proofs can take the form of Merkle paths [50] or note commitments [92] and are evaluated against the cross-chain logic and predefined rules. If these are valid the user can trigger the corresponding transactions on the target chain. The security of this scheme is grounded on the light client implementation in the target chain, which is dependent on the source chain's consensus mechanism. Solutions have been applied to consortium chains [86], to PoW-based chains [50], [128], and to building PoS light clients for the Ethereum 2.0 sync committees ([129]) [44], [51], [91]. Currently, in Ethereum sync committees, a majority of nodes are assumed to be honest, and no slashing mechanism is in place – i.e., accountability for the bridge is not guaranteed[‡]. The potential for relayers to go offline or get compromised poses a risk to the protocol's liveness and safety. When the light client is unsynchronized with the latest state of the source chain, user requests may fail. Additionally, ensuring robust economic security requires the implementation of effective incentivization and slashing mechanisms for relayers. A notable advantage compared to externally verified systems is that even if an entire network of relayers were to collude, they could only disrupt the system if they possess greater mining/voting power than the rest of the source chain, which is equivalent to mounting a 51% attack on that network – this underscores the paramount importance of having secure and resilient source chains.

Validity Proofs. Validity-proof-based bridges rely on proving systems to validate the state of the source chain's consensus mechanism within the target chain [52], [53], [82], [93], [94]. However, contrarily to inclusion proof-based systems, there is no need to understand the exact consensus logic of other ledgers as it only requires verifying a succinct zero-knowledge proof – constant time in zkSNARK-based bridges. The *soundness* of the specific implementation drives the security of these cross-chain bridges. We identify intrinsic drawbacks and limitations associated with the technology. Most ZK proof systems require a trusted ceremony between the prover and the verifier, where a *Common Reference String (CRS)* is generated as a public parameter and used by those parties for proving and verifying proofs [44]. Other protocols do not require a trusted setup but have proven to have efficiency problems, hindering their adoption in practice. In particular, the creation of proofs is the bottleneck. Therefore, techniques such as recursive verification or the parallelization of subcircuits [52] and fine-tune optimizations [44] were proposed. To reduce on-chain operational costs solutions leverage verifiable off-chain computation – i.e., proofs are created by centralized off-chain mechanisms [52], [130]. An optimization technique is to batch multiple blocks and generate single proofs [52], [53]. Moreover, circuits are tailor-made to each specific program, highlighting the lack of flexibility of the technology. ZK schemes are currently unsuitable for widespread adoption in resource-constrained devices [53]. Nevertheless, with further research, ZKP will hold substantial promise and relevance in this domain. There is a pressing need for continued research to enhance proof generation efficiency, reduce memory demands, and lessen reliance on trusted setups. Notably, in March 2023, two zkEVM projects were launched (Polygon [122] and zkSync [131]), and more have followed (e.g., Scroll [132] and Taiko [133]), foreseen to increase Ethereum's scalability while lowering transaction costs.

Fraud Proofs. Fraud proofs allow for securing a cross-chain protocol using a reactive approach [82]. Block headers or other relevant proofs are assumed to be correct until proven otherwise – i.e., are optimistically accepted [134]. External watchers submit fraud proofs to challenge invalid relayed block headers or transaction batches (in optimistic rollups [135]). Before these periods elapse, transactions based on this information are not considered final – i.e., user requests are denied. Watchers are rewarded by presenting fraud proofs, and at the same time, relayers/operators see their stake slashed in case of forwarding invalid block headers or invalid transaction batches [136] – relayer accountability is guaranteed. Safety-wise, there must be a correct watcher online at all times, which is usually assured by the project maintainers [108]. Additionally, choosing an appropriate time window is critical. A usual practice to avoid relying on synchronous communication between parties is to set extended time windows – seven-day periods for settlement in Ethereum [108]. Consequently, a *cctx* takes longer to settle but incurs lower maintenance costs due to most of the blocks not being challenged by watchers.

4.4.4. Secret-based and time-based locks. Hash Time-Locked Contracts (HTLC) [10], [38] is a decentralized asset exchange protocol. It assembles a commit-reveal scheme based on hash-locks and timelocks. Parties agree on parameters off-chain and have predefined periods in which they must act to complete the protocol – i.e., rely on synchronous communication between parties through trusted off-chain communication channels. Both parties are assumed to have read/write access to both ledgers. Therefore, no trust in intermediaries to relay information exists. HTLCs do not guarantee atomicity under longstanding crashes due to the synchronous nature of the protocol. The majority of the solutions alter the synchronous communication assumptions by inserting intermediary networks [80], [87], or focus on the usage of *premiums* [40], [95], [137]. A *premium* is a value staked as collateral before the execution of the actual protocol. It must be a value acceptable by the victim as a

---

[‡] https://github.com/ethereum/consensus-specs/issues/3321

possible compensation for locking up assets for the duration of the protocol. Simultaneously, it needs to be small enough so that parties engage in the swap – i.e., accept the risk of losing this value. There are multiple game-theoretical analyses of HTLCs or simple variations such as [38], [40], [99], [138]. In particular, [138] proves that the protocol is more likely to be completed under collateralized models (i.e., using premiums). However, the actual exchange of premiums (before the protocol) is still vulnerable to attacks [103] (even though these are usually much smaller amounts than the values to swap).

Relying on explicit time intervals is challenging when each permissionless blockchain has different time management mechanisms, usually implemented at a very coarse grain level – in the order of hours or days. Therefore, primitives such as *Verifiable Timed Commitments* (VTC) [139] or *Verifiable Timed Signatures* (VTS) [140] were proposed. In the former, if one party decides not to reveal the value behind the hash commitment, it can be brute-forced by the victim in a configurable number of computation steps. Manevich et al. [106] extends the solution with ZK cryptography to prove arbitrary attributes for the timed commitment. In the latter, parties share signatures from jointly signed refund transactions, which allows one to abort a swap if no action is performed within the agreed duration using the brute force algorithm. The authors of [100] also present a commit-reveal scheme for atomic swaps based on adaptor signatures – verifiable partial signatures that allow revealing a secret once the full signature is published. We question the liveness guarantees of the protocol if one party halts participation midway.

5. Privacy Model for Cross-Chain Systems

Most permissionless blockchains typically rely on unencrypted ledgers and pseudonymous addresses, which have proven to offer limited confidentiality and anonymity [141]–[145]. With data analysis tools, it is possible to track transactional data and map those to real identities [145]. Some initiatives have begun to address this issue by 1) developing privacy-preserving applications deployed to existing networks [146], or 2) launching entirely new blockchains with a strong emphasis on transaction unlinkability and on-chain confidentiality [147]–[149]. A noticeable fact is that many widely used blockchains, including Ethereum, lack these privacy properties by design, leaving users with suboptimal privacy levels.

Privacy and security are frequently intertwined, and in many circumstances, a secure system helps to safeguard its users' privacy. Likewise, guaranteeing privacy might also enhance a system's security – for example in the form of fairness. It is crucial to remember, however, that privacy and security are not synonymous, and one does not guarantee the other. A system may be safe in terms of avoiding unauthorized access but collecting and using personal information in ways that violate rights to privacy.

We break down cross-chain privacy into the privacy of bridge operators (at the IM level) and users [43]. As seen in previous sections, operators play a crucial role in interoperability solutions. Their actions drive the security of the network and its end users. As the public keys of operators are disclosed they can become a target for coercion or resource exhaustion attacks [43]. To neutralize these vulnerabilities, we highlight the importance of hiding identities (i.e., public keys) of operators through the use of privacy-enhancing technologies [127]. As for user privacy, we deconstruct it into transactional data privacy and identity privacy – i.e., the confidentiality of *cctxs* and anonymity of users. As for the first, data associated with cross-chain transactions might be sensitive and can reveal much about the entities holding the data [76], [147], [150]. For the latter, solutions should uphold anonymity while enabling the identification or appropriate punishment of actors engaged in malicious activities. Striking this balance is vital: while privacy is essential, accountability is equally necessary to deter and penalize misconduct. Additionally, privacy is also mandatory to maintain fairness in cross-chain systems. Any interoperability mechanism can influence relaying, accepting or ordering of *cctxs* [151] or can link local transactions issued on different chains.

Our comprehensive literature survey shows that privacy within cross-chain solutions is a relatively understudied domain. In this section, we put forward the first definition and formalization of generic cross-chain privacy by decomposing it into three relevant properties and presenting a taxonomy of privacy-preserving techniques for cross-chain systems. We also discuss some vulnerabilities and attacks that threaten privacy in this domain.

5.1. Privacy Properties

In this section, we formalize three relevant properties of cross-chain privacy-preserving systems: **unlinkability** (of *cctxs*), **anonymity** (of users and operators) and **confidentiality** (of transactional data).

Unlinkability. Single-chain unlinkability is tied to the traceability of funds in transactions issued by the same or related addresses within the same blockchain. Unlinkability must guarantee the actions performed by the same user in the network are not related to each other and that it is not possible to infer properties of those actions from one another – i.e., an account $\mathcal{A}$ and a transaction t are said to be unlinked if it is not possible to infer that $\mathcal{A}$ produced t based on the information available to the observer, for example, through other transactions issued by $\mathcal{A}$. We extrapolate this definition of unlinkability to a cross-chain scenario: where it is not possible to link a transaction in a source blockchain to a transaction in a target blockchain (e.g., lock-mint; cross-chain contract call), or link the addresses that issued those transactions.

Definition 6 (Cross-Chain Unlinkability)**.** *Consider a cctx between two related accounts* $\mathcal{A}_1$ *and* $\mathcal{A}_2$*, where* $\mathcal{A}_1$ *might be equal to* $\mathcal{A}_2$*. Transactions* t *and* t' *issued by* $\mathcal{A}_1$ *and* $\mathcal{A}_2$*, on the source and destination chain, respectively, are said* *to be unlinked iff an external party cannot infer that t and t' are related to each other.*

External parties can infer relationships between transactions using pre-trained models and heuristics which can be queried using functions $f(t, t')$ that return a similarity factor γ – i.e., the probability of two transactions being linked. Heuristics can be related to transaction amounts, certain types of asset profiles, reused addresses, transaction patterns, payment of gas fees, and so on. It should be noted that a cross-chain protocol fundamentally needs linkability to allow for a transaction on one chain to be executed based on a transaction on another. However, it is crucial that this linkability is not observable from an external perspective to protect users' privacy. Trust must be placed in third-party entities or cryptographic mechanisms to guarantee these properties. We elaborate upon this in this section.

Anonymity. Unlinkability in a blockchain is tied to guaranteeing users' anonymity or pseudonymity. An example of pseudonymity is Bitcoin, where users sign transactions with their private key and thus are identified by their public key – i.e., if one can identify the real-world identity behind such key, it is possible to link all the transactions in which it was involved. We provide cross-chain pseudonymity and anonymity definitions.

Definition 7 (Cross-Chain Pseudonymity)**.** *Pseudonymity of $\mathcal{A}$ holds when $\mathcal{A}$ cannot be linked to transactions $t_1, ..., t_k$ it has produced in both ledgers, however, any t_i and $t_j, \forall i, j \in [1, k]$ can be linked to one another.*

Definition 8 (Cross-Chain Anonymity)**.** *Anonymity of $\mathcal{A}$ holds iff 1) $\mathcal{A}$ cannot be linked to transactions $t_1, ..., t_k$ it has issued in both ledgers and 2) t_i and $t_j, \forall i, j \in [1, k]$ are cross-chain unlinkable.*

Confidentiality. Data associated with cross-chain transactions might be sensitive and can reveal much about the entities holding the data. Confidentiality guarantees that critical information is not disclosed and accessible to external entities in or off the network; buyer-supplier relationships in supply chain management systems and medical records are some examples of applications [76], [147], [150].

Definition 9 (Cross-Chain Confidentiality)**.** *Cross-chain confidentiality holds iff the content of any cross-chain transaction $cctx_1$ issued by an address $\mathcal{A}_1$ is indistinguishable from the content of any other cross-chain transaction $cctx_2$ issued by $\mathcal{A}_1$ or any other address.*

The notion of indistinguishability we are trying to capture is similar to IND-CPA: given two cross-chain transactions $cctx_1$ and $cctx_2$, and their raw payloads p_1 and p_2, respectively, an adversary cannot guess which payload p corresponds to each $cctx$ with a probability higher than 50% (i.e., randomly).

5.2. Privacy-Preserving Approaches

In this section, we summarize the main privacy-preserving techniques in the literature, to guarantee at least one of the identified properties.

5.2.1. Zero Knowledge Proofs. Mixing services were the first solutions to break the linkability of transactions in blockchains using zero-knowledge technology. Similarly, they can facilitate guaranteeing *cctxs*' unlinkability [82]. Multiple users transfer funds to a smart contract, which triggers transactions to one or multiple destination addresses that account for the same amount. To withdraw funds, users provide zero-knowledge proof attesting that a certain amount was previously deposited into the contract. A centralized IM can function as transaction mixer [77], [83], [89] – however, this approach can risk user privacy as these links are known to the IM [11], [152]. A solution is multiple chained mix services, providing unlinkability in each hop [153]. Alternatively, the IM can function normally and deposit funds in a mixing contract in the target chain, or to shielded addresses [93], [154]. Transaction mixers incur higher overhead and higher mixing and transaction fees. It is worth noting that, primarily, these techniques are employed to obfuscate traces of illicit activities and launder money obtained through, for instance, cross-chain attacks. Various projects offering mixing services have faced government sanctions – e.g., from the Office of Foreign Assets Control (OFAC) in the US – for their role in facilitating money laundering originating from online criminal activities [155].

ZKP can also help guarantee confidentiality and unlinkability by proving actions without disclosing actual transactions [156]. Proofs can validate that coin commitments are well-formed and not double-spent [92]. ZKP can be used by internal or external mechanisms to assert that the transactional data respects the defined cross-chain rules, without disclosing the parties involved, transaction amounts, or exchange prices [83], [96]. To guarantee a tradeoff between confidentiality and accountability, ZKP might also be used to prove that auditors can decrypt a commitment if there is suspicious activity [94]. This tradeoff is worth exploring in future work. The authors do not explain who would take the auditor role, which is crucial to evaluate the solution. It would be possible to decrypt every *cctx* if colluding entities control all auditors. Interoperability protocols based on permissioned blockchains usually require a trusted IM that needs to access the internal state of each network. Leveraging ZK, one can prove transactions without giving access to the internal state of private chains [89].

5.2.2. Blind Signatures. In cross-chain, signature-based protocols allow extending atomic exchanges to ledgers without scripting capabilities [42], [105]. Blind signatures (BS), initially proposed in [157], allow one user to obtain a signature from a third party such that the third party does not learn anything about the message. Each produced signature represents the same value. Therefore, the third party generates N signatures for a user that escrows N tokens. IMs can issue blind signatures to users. Users present the blind signature in the target chain, and if it is correct, corresponding actions are performed [105]. Since every blind signature is worth the same, BS guarantees *cctxs* unlinkability and

TABLE 2. CLASSIFICATION OF PRIVACY-ENABLER APPROACHES IN BLOCKCHAIN INTEROPERABILITY STUDIES

Privacy Approach	References	# (and %)
PA_1 Zero Knowledge Proofs	[81]–[83], [89], [92]–[94], [96]	8 (47%)
PA_2 Trusted Execution Envir.	[71], [76], [77]	3 (18%)
PA_3 Adaptor Signatures	[100], [104]	2 (12%)
PA_4 Blind Signatures	[105]	1 (06%)
PA_5 Ring Signatures	[43]	1 (06%)
PA_6 Homomorphic Encryption	[101], [102]	2 (12%)

Note: The table categorizes multiple privacy-enabler approaches (PAs) in blockchain interoperability studies. The first column classifies the approach. The second column cites studies or implementations that employ the particular approach. The right-most column estimates the number and percentage of studies adopting each method.

anonymity since one cannot link transactions on both chains (not even the IM). Even though the literature does not explore the application of BS to other interoperability modes besides asset exchanges or permissioned environments, we find relevance in this research direction. In permissioned blockchains, BS ensures confidentiality, unlinkability, and anonymity of *cctxs*. BS promotes fairness through shielding against centralized surveillance, removing the possibility of custom ordering. IMs can still censor by not issuing BS to some addresses. Appendix G.1 presents an asset transfer protocol using BS.

5.2.3. Ring Signatures. Ring signatures provide set anonymity – users are hidden among a ring of k users, where the probability of one having its identity disclosed is *1/k* [158]. However, since ring signatures do not rely on a trusted centralized server, anonymity revocation is impossible – i.e., if there is misbehaviour, that party is not identifiable. Any group of users form a ring by themselves without additional setup, which makes the solution immediately applicable without any previous setup. We acknowledge the existence of different ring signature protocols with trade-offs between, for example, traceability, anonymity and linkability [159].

Ring signatures have been used in privacy-preserving blockchains such as Monero [160] to protect the stealth addresses of users when issuing/receiving transactions. Monero, for example, blends the sender's identity with a set of other identities and generates an unspent transaction to a unique one-time address. In cross-chain protocols, solutions can be similar to the ones proposed in Group Signatures (cf. Section G.3) – an address is blended among a ring of other user addresses that wish to bridge assets to another chain. Since there is no way of determining the signer (assuming a vanilla Ring Signature protocol), the destination address needs to be in the transactional data in the source chain. An IM sends proof to the target chain, which is used to mint the corresponding asset. The problem with this approach is that it makes both transactions linkable due to sharing that destination address – which does not guarantee unlinkability and anonymity. Other variations, such as Verifiable Ring Signatures [161], can offer different guarantees.

Interestingly, our research yielded limited results on cross-chain protocols reliant on Ring Signatures. Among the few that employ this primitive, Wanchain [127] uses it to conceal the transaction sender address, and Bool Network [43] leverages it to shield the identities of the selected committee members, safeguarding against potential threats like DoS attacks. Notably, the committee members rotate constantly, protecting against cryptographic key compromises and guaranteeing *perfect forward secrecy*. However, this approach safeguards the operators but does not protect the users, who maintain the anonymity level provided by the underlying chains.

5.2.4. Adaptor Signatures. Adaptor signatures allow one party to generate a *pre-signature* on a message associated with a secret, which is guaranteed to provide the secret once the full signature is published [104]. It resembles an atomic reveal scheme (ARS) [100] (also called commit-reveal scheme) that underlies HTLCs for asset exchanges. The authors show that an adaptor signature protocol implies an ARS protocol for two parties and thus enables cross-chain atomic swaps. The overall idea is that when party A commits a transaction to collect party B's assets, it reveals a secret that allows B to also A's assets. An alternative to adaptor signatures is running a simple Diffie-Hellman key exchange protocol, which also avoids publishing the shared secret hash that makes HTLCs not guarantee unlinkability. In either approach, applied to permissionless blockchains, it is possible to analyze on-chain transaction amounts. However, it is unlikely that one can link transactions without knowing the cryptocurrencies exchanged and the exchange rate agreed upon off-chain by both parties.

5.2.5. Homomorphic Encryption. Similarly to Adaptor Signatures, one can use homomorphic encryption (HE) to solve commit-reveal schemes linkability problems. Both [101] and [102] proposed an atomic swap solution based on HE where different secrets are deployed in each chain, attaining transaction unlinkability. While there are existing tools for performing basic operations on encrypted data, further research is required to enable more intricate computations and allow general data transfers while guaranteeing confidentiality. Moreover, these protocols typically come with high on-chain computational costs, heavily influenced by the selected homomorphic functions. Appendix G.2 presents a specific algorithm for atomic exchanges using HE.

5.2.6. Trusted Execution Environments. TEEs allow computation on private data without leaking details outside the secure enclave [120]. For confidentiality, the TEE takes user input, executes pre-defined computation (checking compliance with cross-chain rules [76]), reads from blockchains and outputs transactions accordingly [79]. Ideally, transactions issued to the blockchains will not leak sensitive data input to the TEE. However, even though computation within the TEE is confidential, a *cctx* might not be if one is interoperating two public chains as public transactions on both chains can still be linked. It is unclear how un-

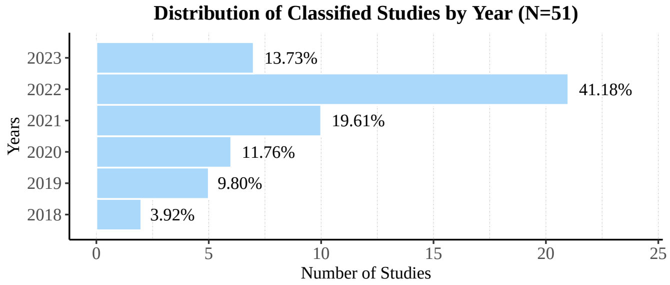


Figure 6. Distribution of classified studies by year. Around 75% of the resources classified in this paper are from 2021 onwards, which highlights the relevance and timeliness of this work.

linkability could be guaranteed for these use cases when the underlying chains do not guarantee confidentiality. We reckon that a possible approach to guarantee unlinkability is to leverage the TEE as a mixing service [77]. However, no previous work provides a specific algorithm to do so. For interoperability solutions supported by evolving notary committees [43] or some solutions with PoS light clients [91], one can use TEEs to protect key-handover procedures. TEEs come with tradeoffs such as additional overhead, limited scalability, and lack of flexibility and composability due to possible vendor lock-in. We note that for asset exchange protocols with confidential order matching algorithms [70] it might be easier to guarantee unlinkability due to the exchange rates not being public.

6. Status Quo of Security and Privacy

In this section, we present the results of our work and extensively discuss the most relevant insights.

6.1. Comparison Framework

We classify 51 academic papers and 6 industry solutions deployed in production (that account for more than 75% of the TVL in cross-chain bridges [25]) in light of the security and privacy models presented in the previous sections. A distribution of the years of the papers classified is depicted in Figure 6. We show that more than half (54%) of the classified papers were published since 2022, which highlights the timeliness of this work. The classification can be seen in Table 3. Additionally, we classify each solution based on a set of performance and usability metrics relevant to both the project maintainers and platforms' users.

6.1.1. Classification Criteria. Next, we classify the relevant IMs according to security, privacy, governance and performance metrics, and miscellaneous properties. For each IM, we attribute security and a privacy approaches, ordered by relevance.

Security Properties.

- ***Integrity (In).*** Integrity is enforced by the underlying cryptographic primitives which are based on the hardness of well-known problems (e.g., computing the discrete logarithm) (●); integrity is enforced under strong assumptions (e.g., trusted hardware, rational participants, parties abiding by laws) (◑); integrity cannot be guaranteed under misbehaving parties (○).
- ***Availability (Av).*** It requires a decentralized network, but there is at least one honest off-chain party (●); availability can be temporarily compromised if any party misbehaves (◑); it is based on a centralized architecture, hence, there are serious concerns over availability (○).
- ***Accountability (Ac).*** The misbehaving party is identifiable and automatically punished (e.g., programmatically) (●); malicious party is identifiable, but there is no punishment or needs to be enforced by a third party (◑); misbehaving parties are neither identifiable nor punished (○) (the notion of accountable safety [91]).

Privacy Properties.

- ***Unlinkability (Un).*** It is cryptographically infeasible to link transactions or addresses (●); it is possible to link transactions or addresses through heuristics (◑); no mechanism is in place to unlink transactions or addresses across domains (○).
- ***Anonymity (An).*** Both users' and operators' identities are concealed (●); users' anonymity or operator's anonymity is enhanced (e.g., through set anonymity approaches) (◑); at most pseudo-anonymity is provided for users, and operators are known (○).
- ***Confidentiality (Cf).*** Data confidentiality is enforced through cryptographic primitives (●); conditional confidentiality – i.e., can be revoked under some circumstances, or verified by auditors. Note that IMs based on private chains partially guarantee this property (◑); there is no confidentiality (○).

Governance and Performance Properties. We extend our classification of solutions with governance [168] and performance [32], [47] properties, as they are factors that intrinsically influence security and privacy [7]. We collect insights from related literature to define:

- ***Decentralization (Dc).*** fully distributed system with a consensus algorithm to settle different views on information [55] or control relies on the end-user (●); limited decentralization of the system, being run by a small set of verifying parties (◑); control of the system resides in less than 4§ parties (can be distributed or centralized) (○).
- ***Latency (Lat).*** Latency of a cross-chain transfer is settled before finalization time (optimistic approach) (●); Latency of a cross-chain transfer is finalized right after the finalization time of the slowest chain (◑); Latency of a cross-chain transfer is more than the finalization time of the slowest chain, due to extra processes ran

---

§. some sources suggest that 4 is a reasonable number of non-colluding parties to secure a blockchain bridge [25].

TABLE 3. CLASSIFICATION OF BLOCKCHAIN INTEROPERABILITY STUDIES IN ACADEMIA AND INDUSTRY.

			Security				**Governance and Performance**			**Privacy**				**Misc.**		
	Ref	Year	Security Approaches	In	Av	Ac	Dc	Lat	Co	Privacy Approaches	Cf	Un	An	IMode	PC	Impl
Academia	[66]	2019	SA_{11}	◑	○	○	○	◑	◑	–	–	–	–	DT	✓	✓
	[69]	2023	SA_{11}	◑	○	○	●	○	–	–[1]	○	○	○	AT	✓	✗
	[70]	2023	SA_{11}	◑	○	◑	○	○	◑	–[2]	○	○	○	AE	✓	±
	[72]	2023	SA_{11}	◑	○	◑	○	○	◑	–	–	–	–	DT	✓	±
	[74]	2020	SA_{11}	●	○	◑	○	◑	●	–	–	–	–	AE	✓	✓
	[76]	2021	SA_{11}, SA_{12}	○	○	○	○	○	○	PA_2	●	●	–	DT	✗	±
	[73]	2022	SA_{11}, SA_{21}	◑	○	○	◑	–	●	–	◑	○	○	DT	✗	✗
	[67]	2021	SA_{11}, SA_{21}, SA_{22}	◑	◑	○	◑	◑	●	–	–	–	–	DT	✗	✓
	[71][5]	2022	SA_{11}, SA_{22}, SA_{31}	◑	◑	●	○	◑	●	PA_2	◑	○	○	AT	✓	✓[3]
	[48]	2019	SA_{11}, SA_{31}	●	◑	●	◑	◑	–	–[1]	●	●	○	DT	✗	✗
	[68]	2023	SA_{11}, SA_{31}	●	◑	○	○	○	–	–[1]	◑	●	○	AT	✗	±
	[75]	2020	SA_{11}, SA_{41}	◑	◑	○	○	○	–	–	–	–	–	AE	✓	±
	[77]	2021	SA_{12}	◑	○	●	○	○	○	PA_2	●	◑[4]	○	DT	✗	±
	[78]	2021	SA_{12}	◑	○	●	◑	○	○	–	–	–	–	AE	✓	✗
	[79]	2019	SA_{12}	◑	◑	◑	○	○	○	–[2]	○	○	○	AE	✓	±
	[82]	2022	SA_{21}	●	●	○	●	○	○	PA_1	○	◑	◑	DT	✓	±
	[81]	2023	SA_{21}, SA_{32}	●	●	○	●	–	○	PA_1	●	●	●	AE	✗	✓
	[80]	2020	SA_{21}, SA_{41}	●	●	◑	●	◑	●	–	–	–	–	AE	✓	✗
	[84]	2022	SA_{22}	●	●	○	●	–	–	–	–	–	–	AT	✓	✗
	[85]	2019	SA_{22}	●	●	●	◑	○	◑	–	–	–	–	DT	✓	✓
	[88]	2021	SA_{22}	●	◑	○	●	◑	◑	–	–	–	–	DT	✓	✓
	[37]	2021	SA_{22}	◑	◑	○	◑	◑	◑	–	–	–	–	AE	✓	±
	[89]	2023	SA_{22}	●	◑	○	○	◑	○	PA_1	–	●	●	AT	✓	±
	[43]	2022	SA_{22}, SA_{12}	◑	●	●	●	○	○	PA_5	○	○	◑	AT	✓	±
	[86]	2021	SA_{22}, SA_{31}	●	●	◑	◑	○	○	–[1]	●	◑	○	DT	✗	±
	[83]	2023	SA_{22}, SA_{32}	●	●	◑	◑	○	○	PA_1	●	◑	◑	AT	✗	✗
	[87]	2022	SA_{22}, SA_{41}	◑	◑	◑	◑	◑	◑	–	–	–	–	AE	✓	±
	[90]	2022	SA_{31}	●	◑	○	●	◑	–	–	–	–	–	AE	✓	✗
	[91]	2022	SA_{31}	◑	●	○	●	–	◑	–	–	–	–	DT	✓	✓
	[58]	2019	SA_{31}, SA_{21}	◑[6]	●	●	●	◑	◑	–	–	–	–	AT	✓	✓
	[92]	2022	SA_{31}, SA_{21}	◑[6]	○	●	●	◑	○	PA_1	○	●	●	AT	✗	✗
	[50]	2020	SA_{31}, SA_{33}	●	●	◑	●	●	●	–	–	–	–	DT	✓	✓
	[53]	2020	SA_{32}, SA_{22}	●	◑	○	●	○	○	–	–	–	–	DT	✓	✓
	[52]	2022	SA_{32}, SA_{31}	●	◑	○	◑	○	○	–	–	–	–	AT	✓	±
	[93]	2023	SA_{32}, SA_{31}	●	◑	○	●	–	○	PA_1	○	●[7]	●[7]	AT	✓	✓
	[94]	2022	SA_{32}, SA_{31}	●	◑	◑	–	○	○	PA_1	◑	◑	◑	AT	✗	±
	[95]	2021	SA_{41}	●	○	●	●	◑	●	–	–	–	–	AE	✓	✓
	[96]	2021	SA_{41}	●	◑	●	○	○	○	PA_1	●	●	○	AE	✓	±
	[97]	2022	SA_{41}	●	○	●	●	◑	●	–	–	–	–	AE	✓	±
	[98]	2022	SA_{41}	●	○	●	●	◑	●	–	–	–	–	AE	✓	±
	[99]	2021	SA_{41}	●	◑	●	●	◑	●	–	–	–	–	AE	✓	±
	[38]	2018	SA_{41}	●	○	○	●	○	●	–	–	–	–	AE	✓	✗
	[40]	2022	SA_{41}	●	○	●	●	◑	●	–	–	–	–	AE	✓	✗
	[100]	2020	SA_{41}	●	○	○	●	◑	●	PA_3	●	●	○	AE	✓	✗
	[101]	2022	SA_{41}	●	◑	○	●	◑	●	PA_6	●	◑	○	AE	✓	✗
	[102]	2018	SA_{41}	●	◑	○	●	–	–	PA_6	○	◑	○	AE	✓	✗
	[103]	2022	SA_{41}	●	◑	◑	●	◑	●	–	–	–	–	AE	✓	✓
	[42]	2021	SA_{41}	●	◑	◑	●	◑	●	–	–	–	–	AE	✓	✓
	[104]	2022	SA_{41}	●	◑	◑	◑	◑	●	PA_3	○	◑	○	AE	✓	±
	[105]	2022	SA_{41}	●	○	◑	○	◑	◑	PA_4	○	●	◑	AE	✓	✓
	[106]	2022	SA_{41}	●	●	○	●	◑	●	–	–	–	–	AE	✓	✓
Industry	[162]	2023	SA_{11}, SA_{22}	●	◑	◑	●	◑	●	–	–	–	–	AT	✓	✓
	[163]	2023	SA_{11}, SA_{22}, SA_{33}	●	◑	◑	●	○	●	–	–	–	–	DT	✓	✓
	[164]	2023	SA_{11}, SA_{22}, SA_{33}	●	◑	◑	○	○	●	–	–	–	–	AT	✓	✓
	[165]	2022	SA_{11}, SA_{22}	●	◑	◑	●	◑	●	–	–	–	–	AT	✓	✓
	[166]	2023	SA_{22}, SA_{32}	●	○	◑	○	◑	○	–	–	–	–	DT	✓	±
	[167]	2023	SA_{33}	●	●	◑	●	●	●	–	–	–	–	AT	✓	✓
Metric addressed in paper #(and%)				57(100%)	57(100%)	57(100%)	55(96%)	51(89%)	50(86%)		23(40%)	24(42%)	23(40%)			
Metric guaranteed in paper #(and%)				39(68%)	13(23%)	14(25%)	30(55%)	2(4%)	23(46%)		9(39%)	15(42%)	4(17%)		46(81%)	15(39%)

The classification criteria are in Section 6.3.1. The table identifies the interoperability mode used by each study (**IMode**), indicating whether it supports Asset Transfers (AT), Data Transfers (DT), or Asset Exchanges (AE). Additionally, it notes if the solution is independent of privacy primitives in the underlying chains (**PC**) and if an implementation is available (**Impl**). Papers marked as (–) do not focus on the specific property. The last two rows of the table summarize the classification. We present a visual representation of 1) the number of studies addressing each metric and 2) the number of studies classified using ● or ✓.

Security approaches: SA_{11} Centralization; SA_{12} Trusted Computation; SA_{21} Permissionless Network; SA_{22} Permissioned Network; SA_{31} Inclusion Proofs; SA_{32} Validity Proofs; SA_{33} Fraud Proofs; SA_{41} Secret- & Time-based Locks.

Privacy Approaches: PA_1 ZKP; PA_2 TEE; PA_3 Adaptor Signatures; PA_4 Blind Signatures; PA_5 Ring Signatures; PA_6 Homomorphic Encryption.

[1] Guarantees some privacy properties even if no privacy approach is employed, due to the use of private chains and secure communication channels (e.g., TLS).

[2] Guarantees privacy at the application layer, not the cross-chain level. Protects the order matching protocol to guarantee fairness but transactions are published normally in blockchains.

[3] Has several open-source implementations in different technological stacks, enhancing decentralization.

[4] With considerable liquidity in the TEE [156] we can classify it as ●.

[5] One of the few solutions being standardized in reputable standardization bodies [56].

[6] Strong dependency on price oracle. It can be classified as ● if the oracle is robust and decentralized [47].

[7] Provided it has a sufficiently large anonymity set.

before (e.g., special account setup) or thereafter (e.g., extra transactions needed) (○).

- ***Cost (Co).*** there are no protocol fees (for the user); can be run with low-tier commercially available hardware (for IM operator) (●); variable fees depending on search and demand with an upper bound lesser or equal than 1% of the bridged value (for the user); requires at most mid-tier hardware (for IM operator) (◑); variable fees depending on search and demand with more than 1% of the bridged value (for the user); requires above mid-tier and/or specialized hardware for the IM operator (○).

Miscellaneous (Misc.). We provide information that complements our assessment. **IMode** shows the main interoperability mode the IM supports. **PC** indicates if the IM requires a privacy-enhanced chain or permissioned blockchain (✗) to operate in optimal conditions, i.e., a dependency (otherwise, ✓). **Impl.** refers if the project has an open-source implementation and evaluation (✓), a not-open source implementation (±), or no implementation (✗).

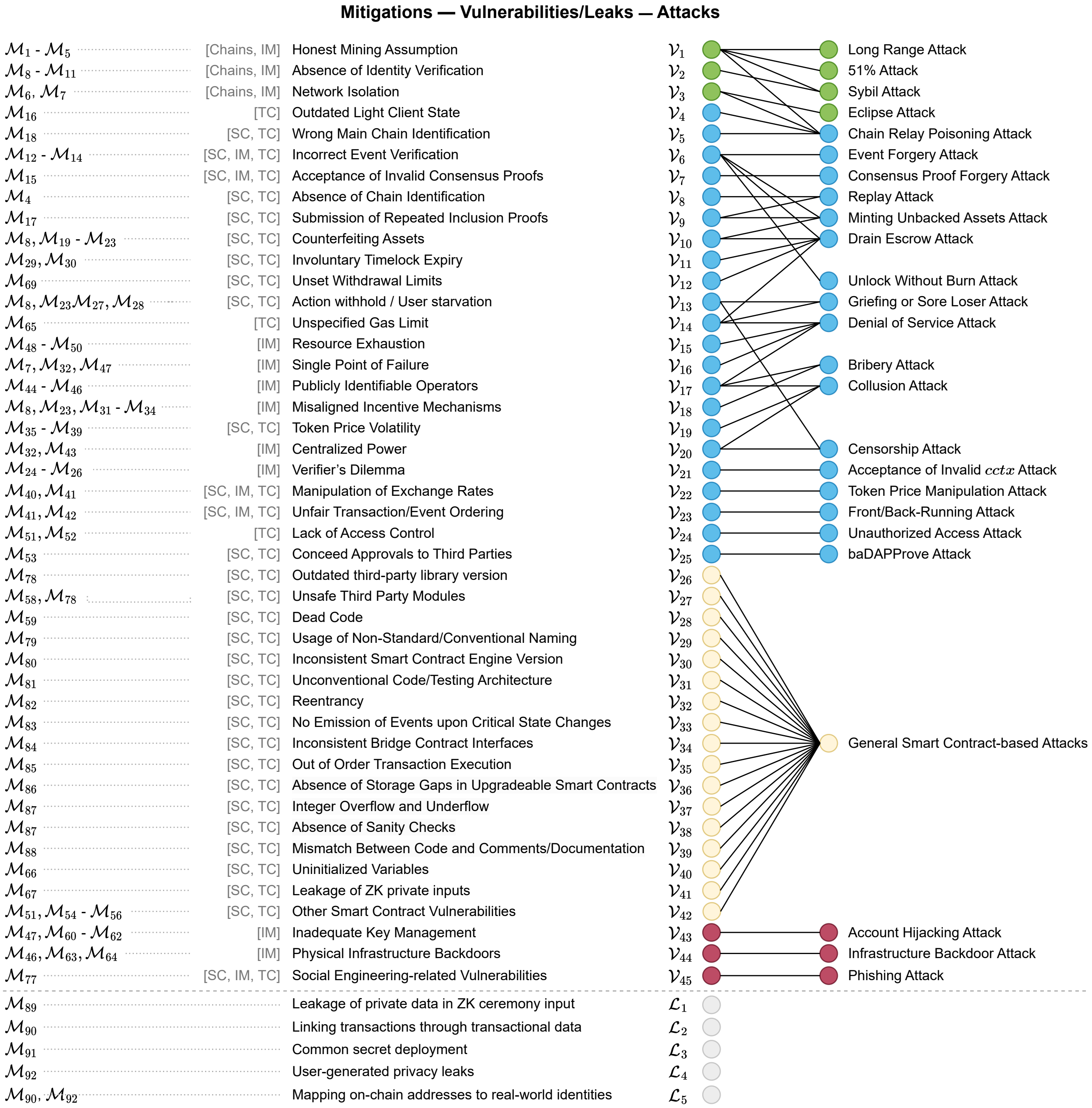
Insights

We now present a list of insights taken from the analysis of the literature.

- **Insight 1:** A conspicuous deficit exists in the literature regarding the empirical assessment of protocol performance and associated costs. This observation is consistent with findings from other studies [7], [169]. While many solutions appear to delegate computationally intensive tasks to off-chain procedures [44], further investigation in this domain remains paramount.
- **Insight 2:** Each study we reviewed ensures a degree of integrity, predominantly upheld by cryptographic mechanisms. It is imperative to meticulously scrutinize these mechanisms. The count of assumptions embedded within a protocol significantly determines its integrity metric. Research that confines its scope to specific adversarial behaviors, such as rational actors, typically exhibits diminished integrity levels.
- **Insight 3:** Within the academic domain, security often takes precedence over privacy, as evidenced by the limited literature addressing privacy properties (15 studies, 30%). In parallel, projects dominating over 75% of the market share seem to neglect cross-chain privacy. This suggests a prevailing apprehension regarding bridge security, relegating privacy to a subordinate design objective.
- **Insight 4:** Zero-knowledge proofs ($\mathcal{PA}_1$) emerge as the predominant approach (50% of IMs with a privacy approach) for guaranteeing privacy in cross-chain systems, which allows shifting trust from third parties to cryptographic protocols. Nonetheless, we believe there is still much work on this front.
- **Insight 5:** The prevailing academic literature primarily emphasizes asset exchanges and transfers between blockchains, with a conspicuous absence of studies addressing general data transfers. This academic trend contrasts with industry developments, which have observed a surge in bridges facilitating data transfers (also called arbitrary message-passing bridges [7]).
- **Insight 6:** For cross-chain anonymity, it is imperative that two distinct transactions remain indistinguishable when originating from the same sender and targeting the same recipient. This condition is attainable exclusively when unique addresses are generated for each *cctx*, epitomized by the mechanism of stealth addresses [154].
- **Insight 7:** The predominant trend in the literature underscores achieving optimal accountability via stake-slashing mechanisms. In contrast, a smaller subset of research advocates for leveraging the legal identification of involved parties, necessitating an ancillary identity service. Such methodologies are predominantly found in centralized frameworks or within permissioned network configurations where nodes possess identifiable attributes.
- **Insight 8:** The privacy dynamics in cross-chain contexts are intricately linked to the privacy paradigms inherent to their foundational domains. Our analysis elucidates that cross-chain unlinkability, in AT and DT-based protocols, can be feasibly realized solely when the foundational chains intrinsically ensure confidentiality. A rigorous formalization of this observation is delineated in Appendix E. For AE protocols, unlinkability can be achieved through cryptographic primitives such as Adaptor Signatures. Additionally, we emphasize the need to research heuristics to break privacy [101], [170] and respective protections against them.
- **Insight 9:** Privacy is expensive. Privacy in interoperability is mostly implemented with a small set of techniques, which bring considerable overhead on the latency and trust assumptions of the protocol. For example [77] is one of the solutions providing confidentiality and unlinkability but relies on trusted hardware.
- **Insight 10:** Asset exchange protocols, with confidential order matching algorithms [70], [79], by themselves, do not offer cross-chain privacy. They protect and ensure the correctness of order matching at the application layer, guaranteeing fairness. However, the actual transactions are public and executed on-chain – i.e., the order matching protocol acts as a fair public forum where people advertise their intentions to buy or sell cryptocurrencies.

6.2. Vulnerabilities, Attacks, and Mitigations on Interoperable Systems

In this section, we present vulnerabilities found in cross-chain. Figure 7 lists and maps each identified vulnerability to corresponding cross-chain attacks and mitigations. Table 6.2 presents all relevant mitigations. Due to its extension, we present the complete explanation of each vulnerability in Appendix H.


Mitigations — Vulnerabilities/Leaks — Attacks
Long Range Attack
51% Attack
Sybil Attack
Eclipse Attack
Chain Relay Poisoning Attack
Event Forgery Attack
Consensus Proof Forgery Attack
Replay Attack
Minting Unbacked Assets Attack
Drain Escrow Attack
Unlock Without Burn Attack
Griefing or Sore Loser Attack
Denial of Service Attack
Bribery Attack
Collusion Attack
Censorship Attack
Acceptance of Invalid $cctx$ Attack
Token Price Manipulation Attack
Front/Back-Running Attack
Unauthorized Access Attack
baDAPProve Attack
General Smart Contract-based Attacks
Account Hijacking Attack
Infrastructure Backdoor Attack
Phishing Attack

$\mathcal{M}_1$ - $\mathcal{M}_5$	[Chains, IM]	Honest Mining Assumption	$\mathcal{V}_1$
$\mathcal{M}_8$ - $\mathcal{M}_{11}$	[Chains, IM]	Absence of Identity Verification	$\mathcal{V}_2$
$\mathcal{M}_6, \mathcal{M}_7$	[Chains, IM]	Network Isolation	$\mathcal{V}_3$
$\mathcal{M}_{16}$	[TC]	Outdated Light Client State	$\mathcal{V}_4$
$\mathcal{M}_{18}$	[SC, TC]	Wrong Main Chain Identification	$\mathcal{V}_5$
$\mathcal{M}_{12}$ - $\mathcal{M}_{14}$	[SC, IM, TC]	Incorrect Event Verification	$\mathcal{V}_6$
$\mathcal{M}_{15}$	[SC, IM, TC]	Acceptance of Invalid Consensus Proofs	$\mathcal{V}_7$
$\mathcal{M}_4$	[SC, TC]	Absence of Chain Identification	$\mathcal{V}_8$
$\mathcal{M}_{17}$	[SC, TC]	Submission of Repeated Inclusion Proofs	$\mathcal{V}_9$
$\mathcal{M}_8, \mathcal{M}_{19}$ - $\mathcal{M}_{23}$	[SC, TC]	Counterfeiting Assets	$\mathcal{V}_{10}$
$\mathcal{M}_{29}, \mathcal{M}_{30}$	[SC, TC]	Involuntary Timelock Expiry	$\mathcal{V}_{11}$
$\mathcal{M}_{69}$	[SC, TC]	Unset Withdrawal Limits	$\mathcal{V}_{12}$
$\mathcal{M}_8, \mathcal{M}_{23}\mathcal{M}_{27}, \mathcal{M}_{28}$	[SC, TC]	Action withhold / User starvation	$\mathcal{V}_{13}$
$\mathcal{M}_{65}$	[TC]	Unspecified Gas Limit	$\mathcal{V}_{14}$
$\mathcal{M}_{48}$ - $\mathcal{M}_{50}$	[IM]	Resource Exhaustion	$\mathcal{V}_{15}$
$\mathcal{M}_7, \mathcal{M}_{32}, \mathcal{M}_{47}$	[IM]	Single Point of Failure	$\mathcal{V}_{16}$
$\mathcal{M}_{44}$ - $\mathcal{M}_{46}$	[IM]	Publicly Identifiable Operators	$\mathcal{V}_{17}$
$\mathcal{M}_8, \mathcal{M}_{23}, \mathcal{M}_{31}$ - $\mathcal{M}_{34}$	[IM]	Misaligned Incentive Mechanisms	$\mathcal{V}_{18}$
$\mathcal{M}_{35}$ - $\mathcal{M}_{39}$	[SC, TC]	Token Price Volatility	$\mathcal{V}_{19}$
$\mathcal{M}_{32}, \mathcal{M}_{43}$	[IM]	Centralized Power	$\mathcal{V}_{20}$
$\mathcal{M}_{24}$ - $\mathcal{M}_{26}$	[IM]	Verifier's Dilemma	$\mathcal{V}_{21}$
$\mathcal{M}_{40}, \mathcal{M}_{41}$	[SC, IM, TC]	Manipulation of Exchange Rates	$\mathcal{V}_{22}$
$\mathcal{M}_{41}, \mathcal{M}_{42}$	[SC, IM, TC]	Unfair Transaction/Event Ordering	$\mathcal{V}_{23}$
$\mathcal{M}_{51}, \mathcal{M}_{52}$	[TC]	Lack of Access Control	$\mathcal{V}_{24}$
$\mathcal{M}_{53}$	[SC, TC]	Conceed Approvals to Third Parties	$\mathcal{V}_{25}$
$\mathcal{M}_{78}$	[SC, TC]	Outdated third-party library version	$\mathcal{V}_{26}$
$\mathcal{M}_{58}, \mathcal{M}_{78}$	[SC, TC]	Unsafe Third Party Modules	$\mathcal{V}_{27}$
$\mathcal{M}_{59}$	[SC, TC]	Dead Code	$\mathcal{V}_{28}$
$\mathcal{M}_{79}$	[SC, TC]	Usage of Non-Standard/Conventional Naming	$\mathcal{V}_{29}$
$\mathcal{M}_{80}$	[SC, TC]	Inconsistent Smart Contract Engine Version	$\mathcal{V}_{30}$
$\mathcal{M}_{81}$	[SC, TC]	Unconventional Code/Testing Architecture	$\mathcal{V}_{31}$
$\mathcal{M}_{82}$	[SC, TC]	Reentrancy	$\mathcal{V}_{32}$
$\mathcal{M}_{83}$	[SC, TC]	No Emission of Events upon Critical State Changes	$\mathcal{V}_{33}$
$\mathcal{M}_{84}$	[SC, TC]	Inconsistent Bridge Contract Interfaces	$\mathcal{V}_{34}$
$\mathcal{M}_{85}$	[SC, TC]	Out of Order Transaction Execution	$\mathcal{V}_{35}$
$\mathcal{M}_{86}$	[SC, TC]	Absence of Storage Gaps in Upgradeable Smart Contracts	$\mathcal{V}_{36}$
$\mathcal{M}_{87}$	[SC, TC]	Integer Overflow and Underflow	$\mathcal{V}_{37}$
$\mathcal{M}_{87}$	[SC, TC]	Absence of Sanity Checks	$\mathcal{V}_{38}$
$\mathcal{M}_{88}$	[SC, TC]	Mismatch Between Code and Comments/Documentation	$\mathcal{V}_{39}$
$\mathcal{M}_{66}$	[SC, TC]	Uninitialized Variables	$\mathcal{V}_{40}$
$\mathcal{M}_{67}$	[SC, TC]	Leakage of ZK private inputs	$\mathcal{V}_{41}$
$\mathcal{M}_{51}, \mathcal{M}_{54}$ - $\mathcal{M}_{56}$	[SC, TC]	Other Smart Contract Vulnerabilities	$\mathcal{V}_{42}$
$\mathcal{M}_{47}, \mathcal{M}_{60}$ - $\mathcal{M}_{62}$	[IM]	Inadequate Key Management	$\mathcal{V}_{43}$
$\mathcal{M}_{46}, \mathcal{M}_{63}, \mathcal{M}_{64}$	[IM]	Physical Infrastructure Backdoors	$\mathcal{V}_{44}$
$\mathcal{M}_{77}$	[SC, IM, TC]	Social Engineering-related Vulnerabilities	$\mathcal{V}_{45}$
$\mathcal{M}_{89}$		Leakage of private data in ZK ceremony input	$\mathcal{L}_1$
$\mathcal{M}_{90}$		Linking transactions through transactional data	$\mathcal{L}_2$
$\mathcal{M}_{91}$		Common secret deployment	$\mathcal{L}_3$
$\mathcal{M}_{92}$		User-generated privacy leaks	$\mathcal{L}_4$
$\mathcal{M}_{90}, \mathcal{M}_{92}$		Mapping on-chain addresses to real-world identities	$\mathcal{L}_5$

Figure 7. Mapping between cross-chain vulnerabilities/leaks, attacks, and corresponding mitigations. For each security vulnerability, we highlight in gray the location where the vulnerability can be found (**Chains** – Underlying Chains; **SC** – Source Chain Smart Contract; **TC** – Target Chain Smart Contract; **IM** – Interoperability Mechanism)

Honest Mining Assumption ($\mathcal{V}_1$). Networks that employ consensus mechanisms with probabilistic finality are subject to forks. A single party that controls more than the security threshold of miners or validators, can authorize invalid *cctxs*, *double-spending* assets [58]. These can be the underlying networks or any intermediary network that serves as IM. Parties holding more power than the predefined security threshold can validate *cctxs* that violate the cross-chain rules.

Absence of Identity Verification ($\mathcal{V}_2$). The absence of identity verification can lead to one party forging multiple identities and gaining more power than perceived in the network [37], [58], [88].

Network Isolation ($\mathcal{V}_3$). Relayers can be intentionally isolated from the rest of the relay network during a period and misled to accept the attacker's chain as the longest [92]. Additionally, packets can be intercepted or dropped causing transactions to not settle in some networks [58], [79]. This can also happen in optimistic-based solutions, even though the attack duration would need to surpass several days or a week, which is practically infeasible [50].

Outdated Light Client State ($\mathcal{V}_4$). Having light clients with outdated information can cause unavailability to respond to SPV requests or inaccurate transaction validation on the destination chain. This can be caused by high relaying costs [58], data unavailability [171], or message delays [69].

Wrong Main Chain Identification ($\mathcal{V}_5$). Relayers can submit conflicting block headers to the target chain smart contract to perform a chain reorganization in the source chain light client [6], [58], [92]. There must be identification mechanisms so that the light client can correctly respond to SPV requests based on the main chain and not on conflicting forks.

Incorrect Event Verification ($\mathcal{V}_6$). Events emitted on blockchains drive interoperability (cf. Section 2.3). The incorrect verification of events might cause the bridge to validate transactions on the target chain based on forged source chain events (or vice versa) [32], [172]–[174].

Acceptance of Invalid Consensus Proofs ($\mathcal{V}_7$). Malicious actors may attempt to construct invalid blocks, not adhering to the consensus rules, or include illegitimate transactions within valid blocks and submit them to the light client implemented in the target chain [130].

Absence of Chain Identification ($\mathcal{V}_8$). One user might try to submit the same proof to multiple destination chains to mint multiple representations of the same locked token, causing to double-spend assets [175].

Submission of Repeated Inclusion Proofs ($\mathcal{V}_9$). Attackers can repeatedly submit the same inclusion proof over and over again to try to prove a statement more than once. Numerous unbacked assets can be created, or multiple tokens can be unlocked, triggered by a single *burn* event [58], [76], [92], [130], [176].

Counterfeiting Assets ($\mathcal{V}_{10}$). Failing to map actions on the destination chain based on events on the source chain may lead to minting assets out of thin air [58], [92], [177].

Involuntary Timelock Expiry ($\mathcal{V}_{11}$). Due to the synchronous nature of some cross-chain protocols, such as HTLCs, parties may incur financial losses if they crash for more than predefined durations [38], [80].

Unset Withdrawal Limits ($\mathcal{V}_{12}$). Cross-chain bridges, especially ones that rely on *lock-mint* patterns, maintain assets in escrow in the source chain, a honeypot for attackers. Multiple attacks have drained such escrows by not setting withdrawal limits based on the usual asset flow [175], [178].

Action Withhold / User starvation ($\mathcal{V}_{13}$). An attacker may intentionally abort a protocol execution or withhold an action to harm other parties or increase the possibility of profitability [40], [74], [78], [95], [106], [179].

Unspecified Gas Limit ($\mathcal{V}_{14}$). Protocols that allow arbitrary message passing are vulnerable to fund draining if gas limits are unset. If relayers execute invalid retryable [180] transactions until success (which will never happen), they will eventually be out of funds.

Resource Exhaustion ($\mathcal{V}_{15}$). Instead of inducing abnormal behaviour in the system, attackers may focus on disrupting the availability of a cross-chain solution, for example, by compromising a centralized interoperability mechanism.

Single Point of Failure ($\mathcal{V}_{16}$). The failure of components that compromise the liveness of a cross-chain solution are single point of failure. These can be oracles used to fetch prices [181], or centralized components in the architecture [175], [182].

Publicly Identifiable Operators ($\mathcal{V}_{17}$). Solutions, where operators are public and identifiable, are vulnerable to multiple attack vectors, as attackers can focus their efforts on attacking powerful entities or organizations securing the system – *Bribery*, *Collusion*, *DoS*, and *Phishing* are some examples [43]. These can compromise both the safety and liveness of a cross-chain bridge.

Misaligned Incentive Mechanisms ($\mathcal{V}_{18}$). Incentivization is paramount in decentralized systems. In the BAR behaviour model [183], Rational players follow strategies that increase their profits – i.e., they might choose to deviate from the protocol rather than following the rules due to the more attractive economic incentives. If protocols do not guarantee attractive rewards, adversaries might be more incentivised to misbehave rather than follow the protocol [50], [77], [82], [138], [184], [185].

Token Price Volatility ($\mathcal{V}_{19}$). Protocols relying on cryptocurrencies suffer from some vulnerabilities inherited from DeFi. One example is the high volatility of token prices. It can lead to unfair trades [43], [58], [92], [95], [97], [98], [124], or compromise the security of a bridge if tokens in escrow suddenly become worthless.

Centralized Power ($\mathcal{V}_{20}$). Centralization must be evaluated across different layers. Protocols can rely on centralized infrastructure [48], [79], [186] or distributed infrastructure but be mainly controlled by a single entity [187] – e.g., centralized governance procedures [188] or centralized computation conducted by L2 bridge operators [66], [108], [187], [189]. Possible consequences are liveness compromise, transaction censorship [82] or transaction reordering [182], [190].

Verifier's Dilemma ($\mathcal{V}_{21}$). The Verifier's Dilemma, initially proposed by [191], shows that rational blockchain miners benefit from skipping the verification of blocks to gain an advantage in proposing subsequent blocks. The probability of such behavior increases when blocks contain computationally expensive transactions. We acknowledge that cross-chain solutions based on third-party networks also suffer from this vulnerability, where a *cctx* might not be fully validated.

Manipulation of Exchange Rates ($\mathcal{V}_{22}$). Token prices from external sources are inserted into blockchains by oracles. Oracles can be manipulated to send an erroneous price feed [22], [192]–[195]. Bridges, or users themselves can therefore use the wrong exchange rates leading to unfair or unrealistic trades.

Unfair Transaction/Event Ordering ($\mathcal{V}_{23}$). Transaction ordering techniques enforced by blockchain miners through MEV are also found in a cross-chain scenario [82]. Similarly to the unfair ordering in the miners' mempool, the interoperability mechanisms that relay block headers, events, or any other type of proofs between blockchains, can also be subject to custom order based on the maximum extractable profit.

Lack of Access Control ($\mathcal{V}_{24}$). With the rapid evolution of decentralized applications' development, the complexity of such apps has increased exponentially. However, the absence of access control policies when accessing certain functionalities (e.g., usually implemented as smart contracts) has originated multiple attacks in these components [196]–[201].

Conceed Approvals to Third Parties ($\mathcal{V}_{25}$). The usage of functions such as *approve()*, *permit()* and *transferFrom()* available in some token standards such as ERC20, make users vulnerable to fund theft [173], [202]. The baDAPProve problem, found in the Multichain bridge hack [203], refers to the users permitting the bridge contract to spend tokens on their behalf to save gas fees. If the bridge gets hacked, all user funds are drained.

Outdated third-party library version ($\mathcal{V}_{26}$). Infrequent third-party library version updates may leave security patches unapplied [204].

Unsafe Third Party Modules ($\mathcal{V}_{27}$). As usual in software development, code relies on third-party modules or libraries. These libraries can insert vulnerabilities into the codebase, which may weaken the source code [172], [175], [181], [182], [201].

Dead Code ($\mathcal{V}_{28}$). A noteworthy vulnerability behind the Qubit and Multichain hack is the presence of dead code within the deployed smart contracts, allowing attackers to execute malicious operations (cf. Table J).

Usage of non-standard/conventional naming ($\mathcal{V}_{29}$). Different programming languages use specific rules. Some examples are naming conventions for the names of variables and functions, or the usage (or not) of curly brackets [181], [204].

Inconsistent smart contract engine version ($\mathcal{V}_{30}$). Multiple audits have found that, within the same project, smart contracts are using different versions of smart contract engines [175], [182], [205].

Unconventional code/testing architecture ($\mathcal{V}_{31}$). Unconventional architectures at the protocol and implementation levels present a challenge to building secure and scalable bridges. At the implementation level, it is difficult for auditors to evaluate the codebase and for practitioners to understand the different components' locations.

Reentrancy ($\mathcal{V}_{32}$). This vulnerability is found when a smart contract calls an untrusted contract, and the latter recursively calls the initial one to manipulate its internal state [175].

No emission of events upon critical state changes ($\mathcal{V}_{33}$). Cross-chain systems revolve around events. Off-chain mechanisms listen for events that indicate state changes and sometimes forward them to other chains. Not emitting [172], [206], or emitting wrong events [182] upon state changes can risk the integrity of the bridge.

Inconsistent bridge contract interfaces ($\mathcal{V}_{34}$). Bridges are composed of multiple components that must communicate with one another through standardized interfaces [207]. Not guaranteeing consistent bridge contract interfaces may cause an indefinite loss of funds, due to messages sent by one party are not understood by the other.

Out of order transaction execution ($\mathcal{V}_{35}$). An auditability to Arbitrum's code has found a vulnerability where an attacker can exploit the absence of an ordering mechanism to deny a user access to its assets [172].

Absence of storage gaps for upgradeable smart contracts ($\mathcal{V}_{36}$). Not following the storage gaps pattern [208] does not allow for inserting new state variables in the future without compromising the storage compatibility with existing deployments.

Integer overflow and underflow ($\mathcal{V}_{37}$). Attempting to store values higher or lower than the largest and least value supported by a data type incurs an overflow or underflow, respectively. This vulnerability might allow an attacker to drain a bridge by convincing the bridge that the value is within the expected range when it is not [172], [178], [182], [204].

Absence of Sanity Checks ($\mathcal{V}_{38}$). Throughout the codebase, there must be checks to ensure the bridge functions as intended, safeguarding its integrity. We provide some examples. Make sure an address received as input is what is expected – i.e., an EOA address, a contract address with a predetermined function [172], [175], [178]), checking function return types [182], operations for arithmetic errors [204], ensuring there are no operations on null addresses [172], [178], [181], inconsistent data type conversions [181], [182], and the size of the payload being transferred in the bridge [204].

Mismatch between code and comments/documentation ($\mathcal{V}_{39}$). Several audits have revealed occasional inconsistencies between the code and its accompanying comments [175], [205], [206] or documentation [181], [182], [204], [205] as they can mislead both developers and auditors.

Uninitialized variables ($\mathcal{V}_{40}$). Uninitialized variables, mainly done to save gas fees [209] can lead to the internal state believing it has not been initialized. Attackers can

TABLE 4. List of Mitigations Collected in the Literature and Proposed by our analysis (marked with –)

Label	Ref	Mitigation description
$\mathcal{M}_{1}$	[90]	Wait full confirmation time according to the source chain consensus mechanism
$\mathcal{M}_{2}$	[58]	Insertion of block maturity periods
$\mathcal{M}_{3}$	[55]	Usage of blockchain views
$\mathcal{M}_{4}$	[175]	Add chain identification mechanisms
$\mathcal{M}_{5}$	[58]	Synchronize smart contract state on multiple destination chains
$\mathcal{M}_{6}$	[50]	Increase transaction settlement time
$\mathcal{M}_{7}$	[66]	Physical decentralization of infrastructure
$\mathcal{M}_{8}$	[70]	Usage of a trusted centralized authority to mediate *cctxs*
$\mathcal{M}_{9}$	[210]	Integration with Self Sovereign Identity (SSI) mechanisms
$\mathcal{M}_{10}$	[37]	Make the creation of identities expensive (e.g., a high stake per identity)
$\mathcal{M}_{11}$	[88]	Reward creating fewer identities with more stake
$\mathcal{M}_{12}$	[173]	Listen to events only from whitelisted smart contracts
$\mathcal{M}_{13}$	[174]	Deploy runtime monitoring modules
$\mathcal{M}_{14}$	[82]	Employ multiple different monitoring strategies at the same time
$\mathcal{M}_{15}$	[91]	Enable verifiability of state updates in light clients for different consensus mechanism
$\mathcal{M}_{16}$	[171]	Insertion of a data availability layer
$\mathcal{M}_{17}$	[92]	Unique nonce/id generation per request
$\mathcal{M}_{18}$	[58]	Use and develop new main chain identification mechanisms
$\mathcal{M}_{19}$	[92]	Trigger automatic liquidations of collateral
$\mathcal{M}_{20}$	[58]	Use Collateralization / Over-Collateralization techniques
$\mathcal{M}_{21}$	[135]	Usage of external incentivized watchers that attest actions/events
$\mathcal{M}_{22}$	[188]	Embedded rules in third party network consensus mechanism
$\mathcal{M}_{23}$	[103]	Usage of Distributed Signature Schemes between untrusted users and operators
$\mathcal{M}_{24}$	[190]	Parallelizing transaction verification
$\mathcal{M}_{25}$	–	Insert independent computational-heavy transactions into multiple blocks
$\mathcal{M}_{26}$	[211]	Separate entities that create and verify blocks
$\mathcal{M}_{27}$	[95]	Usage of Premiums
$\mathcal{M}_{28}$	[211]	Usage of Verifiable Timed Commitments
$\mathcal{M}_{29}$	[80]	Provide support for periods of asynchrony in the execution of the protocol
$\mathcal{M}_{30}$	[42]	Use pre-deployed refund transactions/contracts triggered upon failures
$\mathcal{M}_{31}$	[138]	Model and analyze user behaviour through game-theory principles
$\mathcal{M}_{32}$	[186]	Protocol architecture decentralization
$\mathcal{M}_{33}$	[50]	Increase the number of parties and scatter mining power among them
$\mathcal{M}_{34}$	[99]	Usage of MEV to front-run misbehaving transactions
$\mathcal{M}_{35}$	[97]	Parallel asset locking
$\mathcal{M}_{36}$	[97]	Reduce time window for users to observe price fluctuations
$\mathcal{M}_{37}$	[58]	Over-collateralization to account for slippage
$\mathcal{M}_{38}$	[58]	Adjust the amount locked according to the updated exchange rates
$\mathcal{M}_{39}$	[58]	Trigger automatic liquidations to avoid getting uncollateralized
$\mathcal{M}_{40}$	[186]	Merge multiple sources of data
$\mathcal{M}_{41}$	[212]	General mitigations for (MEV), such as confidential mempools
$\mathcal{M}_{42}$	[213]	Enforce predefined transaction ordering rules
$\mathcal{M}_{43}$	[190]	Overlap capabilities between multiple parties
$\mathcal{M}_{44}$	[43]	Employ evolving committees rather than static ones
$\mathcal{M}_{45}$	[214]	Hide one public key among multiple keys of other users/operators
$\mathcal{M}_{46}$	[66]	Other usual web2 infrastructure backdoor mitigations
$\mathcal{M}_{47}$	[66]	Decentralization at the operational level (e.g., key management and monitoring)
$\mathcal{M}_{49}$	[48]	Use redundant nodes or deploy logic in the blockchain (i.e., in smart contracts)
$\mathcal{M}_{50}$	[66]	Usual web2 practices (e.g., rate limiting, challenge-response tests)
$\mathcal{M}_{51}$	–	Multiple rounds of smart contract audits, preferably by different parties
$\mathcal{M}_{52}$	[56]	Standardization of cross-chain bridge design (e.g., for proper access control)
$\mathcal{M}_{53}$	[203]	Do not issue approvals for more funds than what is strictly necessary
$\mathcal{M}_{54}$	[215]	General smart contract vulnerabilities mitigations
$\mathcal{M}_{55}$	–	Submit codebases to thorough code reviews before production
$\mathcal{M}_{56}$	–	Ensure there are rigorous testing guidelines being enforced
$\mathcal{M}_{57}$	–	Just like on-chain smart contracts, off-chain programs and infrastructure must be audited
$\mathcal{M}_{58}$	[182]	Avoid library version auto-upgrades and audit code before upgrading
$\mathcal{M}_{59}$	–	Linting tools to raise warnings for unused code
$\mathcal{M}_{60}$	[216]	Improve cryptographic key management (e.g., usage of hardware or cold wallets)
$\mathcal{M}_{61}$	[77]	Increase of the number of validators and thresholds in multi-signature wallets
$\mathcal{M}_{62}$	[66]	Employ further authentication mechanisms to protect keys
$\mathcal{M}_{63}$	[66]	Accept incoming connections only from whitelisted IP addresses
$\mathcal{M}_{64}$	[66]	Authenticate requests made to RPC nodes through rotating keys
$\mathcal{M}_{65}$	[180]	Require setting gas limits for *cctxs*
$\mathcal{M}_{66}$	–	Performe deep optimizations once the industry and the project have reached stability
$\mathcal{M}_{67}$	[44]	Dispose of private inputs used to generate the CRS in zk-based solutions
$\mathcal{M}_{68}$	[32]	Monitor on- and off-chain infrastructure
$\mathcal{M}_{69}$	[188]	Set appropriate withdrawal limits and implement a freezing functionality
$\mathcal{M}_{70}$	[203]	Do not give excessive permissions to individual external entities
$\mathcal{M}_{71}$	[200]	Check inputs in arbitrary message passing bridges for function signatures' hash collision
$\mathcal{M}_{72}$	–	Treat critical fixes internally before pushing them to public repositories
$\mathcal{M}_{73}$	–	Make sure critical components are updated before an audit, not afterwards
$\mathcal{M}_{74}$	–	Do not launch projects on top of existing ones without knowing the inner details
$\mathcal{M}_{75}$	–	Fix bugs as soon as they are detected, not just leaving for the future
$\mathcal{M}_{76}$	[7]	Follow standard practices, such as RFCs.
$\mathcal{M}_{77}$	–	Increasing the awareness of all involved actors
$\mathcal{M}_{78}$	[204]	Attest the security of external packages using analysis tools and third-party auditors
$\mathcal{M}_{79}$	[205]	Follow coding practices according to the programming language being used
$\mathcal{M}_{80}$	[182]	Apply the same (or compatible) compiler version across the whole project
$\mathcal{M}_{81}$	[181]	Follow standard code/testing architectures to prioritize understandability of the code
$\mathcal{M}_{82}$	[175]	Update the internal state of a contract before making an external call to another one
$\mathcal{M}_{83}$	[182]	Document critical state changes and – e.g., one event should be emitted for each one
$\mathcal{M}_{84}$	[207]	Reuse code for the definition of messages for components that interact with one another
$\mathcal{M}_{85}$	[172]	Enforce transaction ordering between L1s and L2s
$\mathcal{M}_{86}$	[175]	Follow standards for the usage of storage gaps within upgradeable smart contracts
$\mathcal{M}_{87}$	[217]	Use (e.g., static) analysis tools to warn the absence of checks on inputs and operations
$\mathcal{M}_{88}$	[206]	Force documentation and comments to be updated once pull requests are accepted
$\mathcal{M}_{89}$	–	Providing a user-agnostic and random string as input for the ZK trusted ceremony phase
$\mathcal{M}_{90}$	[160]	Use unique addresses – e.g., using primitives such as stealth addresses
$\mathcal{M}_{91}$	–	Rely on alternative atomic-reveal schemes – e.g., Diffie Hellman and Adaptor Signatures
$\mathcal{M}_{92}$	[141]	Educate users for privacy-preserving practices – e.g., address reuse and unique gas prices

Note: The table displays various security and privacy vulnerability mitigations. We have included references to indicate the source of each vulnerability and marked our proposals with "–".

then initialize the contract by passing attacker-controlled addresses as whitelisted contracts.

Leakage of ZK private inputs ($\mathcal{V}_{41}$). As introduced in Section 4.4.3, the CRS used to create and verify ZK proofs is computed using private inputs provided to an MPC scheme. The leakage of these inputs can lead to an adversary being able to forge proofs and generate cross-chain state transitions that violate the defined cross-chain rules.

Other Smart Contract Vulnerabilities ($\mathcal{V}_{42}$). We do not explore all smart contract-related vulnerabilities due to their extension. Rather, we point the reader to an extensive work surveying vulnerabilities in this context [215]. We present some bridge-related vulnerabilities. These range from signature verification bypass in the Wormhole hack [205], incorrect usage of modifiers [172], [182], unauthorized smart contract calls in the first PolyBridge hack [200] and wrong function visibilities [182].

Inadequate Key Management ($\mathcal{V}_{43}$). The compromise of cryptographic keys is one of the main sources of hacks in cross-chain bridges [110], [173]. Even worse than compromising a single key, is compromising multiple keys, which has happened more than once (cf. Table J).

Physical Infrastructure Backdoors ($\mathcal{V}_{44}$). Infrastructure backdoors create numerous potential attack vectors, such as reaching blockchain nodes through the RPC or HTTP ports which can be used to transmit malicious transactions or perform DDoS attacks [66].

Social Engineering-related Vulnerabilities ($\mathcal{V}_{45}$). Attacks such as *Phishing* or *Ransomware Attacks* can be performed through social engineering practices, usually in social media or untrusted websites [202], [214].

6.3. Real World Cross-Chain Bridge Hacks

Attacks against cross-chain bridges have proliferated in the last couple of years. Table 5 presents a classification of 14 of the most impactful attacks in the industry since July 2021, that account for more than 94% of the total value stolen from cross-chain bridges (cf. Table J).

6.3.1. Classification Criteria.

We present general attack information, incident response-related data, the components targeted by the attackers, and the vulnerabilities behind each. In the table, we mark as *No Information Available (–)* the data points to which we could not gather data – i.e., the corresponding team did not respond or provide the data. Appendix J presents further information and mitigations for each. Interestingly, the number of cross-chain hacks plummeted in 2023, which can be explained by the drop in token prices during the bear market [230].

Security Approach (SA). The security approach used by the bridge.

Date. The date of the first transaction exploiting a vulnerability in the protocol.

Amount. The amount in USD stolen from the cross-chain bridge. We don't include any collateral losses in other protocols.

Attacker Type (AT). There may be one or multiple attackers taking advantage of a vulnerability. We classify them as black or white hats based on whether they returned the funds (or both if there is at least one attacker of each type). Attackers that restituted the funds, excluding agreed bounty fees, are also considered white hats in our analysis.

Number of Transactions (Txs). A range of the number of transactions issued by the attackers to exploit the bridge, encompassing both external and internal transactions, which are transactions issued directly by the user or as a consequence of another contract execution, respectively. It does not include transactions issued before or after the attack to exchange or launder funds using DEXes (e.g., Uniswap) or mixing services.

Usage of Mixers (Mix). The usage of transaction mixers (e.g., Tornado Cash) by the attacker to launder funds either before or after the attacks to break the linkability of transactions.

Discovery Time (DT). The time it took maintainers to discover the attack and trigger the corresponding incident response mechanism. Given that this information is internal to each team, we asked all 14 projects to provide us with data.

Communication Time (CT). The time it took maintainers to communicate the exploit to the community. This communication was performed solely as *Tweets*. This value is the difference between the timestamp of the *Tweet* and the timestamp of the first exploit transaction.

In terms of vulnerabilities, our classification encompasses both the vulnerabilities associated with each attack and the specific components of our model where these vulnerabilities were found. We also found it important to note that in some cases, funds may be taken from different components than where the vulnerability exists.

Vulnerability Location (VL). We identify the location of each vulnerability (cf Section X). Possible locations are: in the *Source Chain Smart Contract* – the component with the bridging logic in the source chain, responsible for escrowing funds; in the *Target Chain Smart Contract* – the element with the bridging logic in the source chain, responsible for verifying inclusion proofs; or in the *Interoperability Mechanism* – the off-chain component that enables interoperability, usually composed of validators/relayers.

Exploit Location (EL). One vulnerability in one location can originate exploits in others. As an example, in the Ronin bridge hack, compromising the private keys of the off-chain relayers (functioning as *Interoperability Mechanism*) allowed the attacker to unlock funds in the *Source Chain SC*. Therefore, besides *VL*, we classify the exploit location as follows: in the *Source Chain Smart Contract* if the attacker stole escrowed funds; in the *Target Chain Smart Contract* if the attacker minted unbacked funds; or in the *Business Logic Smart Contract* if the attacker stole funds by exploiting the business logic contract – usually because users approved a bridge-controlled contract to manage their funds (e.g., through the *approve()* function in the ERC20 token standard).

TABLE 5. CLASSIFICATION OF MOST PROFITABLE CROSS-CHAIN BRIDGE HACKS GROUPED BY SECURITY APPROACH. THE AMOUNTS ARE PRESENTED IN USD. THE CELLS WITH THE VULNERABILITY NUMBER ARE FILLED WITH THE COLOR ACCORDING TO THE LAYER THEY BELONG TO (CF. SECTION 4.2). WE ADD A "SUMMARY" ROW THAT AGGREGATES INFORMATION. SPECIFICALLY, WE USE CELL SHADING TO SHOW THE PERCENTAGE OF HACKS IN WHICH EACH VULNERABILITY WAS FOUND.

Project Information		**General Attack Information**					**Incident Resp**		**Where**		**Mapping to Theoretical Vulnerabilities**					
Name & Ref	SA	Date	Amount	AT	Txs	Mix	DT	CT	VL	EL	$\mathcal{V}_{44}$	$\mathcal{V}_{43}$	$\mathcal{V}_{28}$	$\mathcal{V}_{27}$	$\mathcal{V}_{24}$	$\mathcal{V}_{6}$
[218] Ronin	$\mathcal{SA}_{22}$	Mar 2022	624M	■	○	◑	6d	●	IM	SC	✓	✓	✗	✗	✗	✗
[219] PolyBridge #1	$\mathcal{SA}_{22}$	Aug 2021	611M	□	◔	○	–	◔	TC	SC	✗	✓	✓	✗	✗	✗
[220] BNB	$\mathcal{SA}_{11}$	Oct 2022	566M	■	◔	◐	–	◑	TC	TC	✗	✗	✗	✗	✓	✗
[123] Wormhole	$\mathcal{SA}_{22}$	Feb 2022	326M	■	○	◐	–	○	TC	TC	✗	✗	✓	✗	✓	✗
[221] Nomad	$\mathcal{SA}_{33}$	Aug 2022	190M	◧	◕	◑	–	◔	SC	SC	✗	✗	✗	✗	✓	✗
[222] BXH	$\mathcal{SA}_{11}$	Oct 2021	139M	■	○	◐	–	◑	–	SC	✓	✓	✗	✗	✗	✗
[223] Multichain #2	$\mathcal{SA}_{22}$	Jul 2023	126M	■	○	○	–	◑	IM	SC	✓†	✓†	✗	✗	✗	✗
[224] Harmony	$\mathcal{SA}_{22}$	Jun 2022	100M	■	◔	◑	–	◕	IM	SC	✓	✓	✗	✗	✗	✗
[225] Qubit	$\mathcal{SA}_{11}$	Jan 2022	80M	■	◔	◑	–	◔	SC	TC	✗	✗	✗	✓	✓	✗
[226] pNetwork	$\mathcal{SA}_{33}$	Sep 2021	13M	■	◔	○	13m	◔	IM	SC	✗	✗	✗	✗	✗	✓
[227] Thorchain #3	$\mathcal{SA}_{21}$	Jul 2021	8M	■	○	◑	–	–	IM	SC	✗	✗	✗	✗	✗	✓
[223] Anyswap	$\mathcal{SA}_{22}$	Jul 2021	8M	■	○	◑	–	◕	IM	TC	✗	✓	✗	✗	✗	✗
[227] Thorchain #2	$\mathcal{SA}_{21}$	Jul 2021	5M	■	◕	◑	–	◑	IM	TC	✗	✗	✗	✗	✓	✓
[219] PolyBridge #2	$\mathcal{SA}_{22}$	Jul 2023	4.4M	■	◑	○	7h	◕	IM	TC	✗	✓	✗	✗	✗	✗
[228] Meter	$\mathcal{SA}_{22}$	Jul 2021	4.4M	■	○	◑	–	◔	SC	TC	✗	✗	✗	✗	✓	✗
[229] Chainswap	$\mathcal{SA}_{22}$	Jul 2021	4.4M	■	●	●	–	◑	TC	TC	✗	✗	✓	✗	✓	✗
[223] Multichain #1	$\mathcal{SA}_{22}$	Jan 2022	3M	◧	–	●	–	◕	TC	BL	✗	✗	✗	✓	✓	✗
[227] Thorchain #1	$\mathcal{SA}_{21}$	Jun 2021	140K	■	–	◑	5m	–	IM	TC	✗	✗	✗	✗	✗	✓
Summary		**07/21 - 07/23**	**2.9B**								**25%**	**44%**	**19%**	**13%**	**50%**	**25%**

Attacker Type (AT)
■ Black hat
□ White hat
◧ Black and white hats

Number of Transactions (Txs)
○ 1-10
◔ 10-50
◑ 50-100
◕ 100-1000
● >1000

Usage of Mixers (Mix)
○ Not used
◐ Before the attack
◑ After the attack
● Before and after the attack

Communication Time (CT)
○]0; 2] hours
◔]2; 4] hours
◑]4; 6] hours
◕]6; 24] hours
● >= 6 days

Vulnerability/Exploit Location (VL/EL)
SC Source Chain SC
TC Target Chain SC
IM Interoperability Mechanism
BL Business Logic SC

– No information available / Team did not respond

† Still to be confirmed

Discovery Time (DT)

Insights

We present a list of insights taken from the analysis of cross-chain bridge hacks.

- **Insight 1:** 65.8% of the total value stolen originated in bridges associated with SA_{22}. Projects choose SA_{22} to have finer control over the bridge. However, it also eases hackers' efforts to gain control over the infrastructure. 3 hacks were performed on solutions with SA_{11} and SA_{22} (26.8% and 0.5% of the total value, respectively); and two projects based on SA_{33} (6.9%). SA_4 approaches do not enter the leaderboard.
- **Insight 2:** Limiting the number of internal transactions within the same contract or the amount moved per external transaction is possible and advisable. We believe setting withdrawal limits or emergency pauses would significantly reduce this value. This empirical analysis remains a task for future endeavors.
- **Insight 3:** Only one hack is classified as being performed by a white hat due to returning almost all funds. Besides this record, only around 35M USD were returned (1.5% of the hacked amount). There is clearly a lack of motivation for hackers to disclose vulnerabilities. Additionally, as shown in [231], there is little transparency regarding the bounties offered.
- **Insight 4:** Notably, in 14 of the hacks authored by black hats, transaction mixers were used 5 times before the attack (35.7%) and 11 times after the attack (78.6%). The pNetwork hackers did not use any mixer and still retain funds in their addresses [232]. In the PolyBridge hack, the hacker returned a noteworthy portion of the 611M USD after negotiations [233]. We believe these highlight the difficulty of money laundering in blockchain environments compared to conventional theft due to the inherent traceability of blockchain transactions [234].
- **Insight 5:** We find that the *lock-mint* model for asset transfer bridges is riskier than other approaches. Attackers target escrowed funds in the source chain. Eight hacks drained funds from the escrow in the source chain, accounting for 1.8B USD (62%). Using native tokens instead of wrapped assets is a solution allowing developers to implement one-way flows – *burn-mint* models. An example is Circle's USDC announcement in October 2023. USDC is now burned in Ethereum and minted natively on Polygon [235].
- **Insight 6:** The collected data indicates that the Thorchain and pNetwork teams took 5 and 13 minutes, respectively, to detect the incidents. However, the Ronin bridge team took a significantly longer period of 6 days. This emphasizes the need for improvement in achieving fast incident discovery. Furthermore, the substantial amounts stolen from these protocols raise concerns regarding the possibility of implementing withdrawal limits. Such limits could act as a safeguard by halting withdraws if the withdrawal amounts exceed a certain threshold. In the literature, there is little research and information on incident response. We refer the reader to [47], which serves as a foundational paper to decide which interoperability solution to choose, and thus,

realize a possible threat model and appropriate incident response framework. Additionally, the mechanisms to identify incidents need to be thoroughly studied. Work has been done designing cross-chain models to identify and visualize such deviations [32], [174], [236].

6.4. Recommendations to Cross-Chain Bridge Operators

We divide the different guidelines for cross-chain systems into three different domains.

6.4.1. Implementation Level. Like any computer program, smart contracts are vulnerable to attacks [26], [215], [237], [238]. As we speak, attacks originated in vulnerabilities identified long ago have been happening [239]. To address these vulnerabilities, developers must implement secure coding practices, use Continuous Integration practices, and use tools to identify and mitigate potential security issues. Some of these include static analysis (e.g., Slither [217], Mythril [240], Mythx [241]), formal verification [242], fuzz testing (e.g., Echidna [243], Harvey [244]), vulnerability detection at runtime (e.g., Scribble [245]), and more recently AI tools to identify vulnerable patterns and perform analysis of control/data flow graphs [26]. These are run directly against the codebase or at the bytecode level. Properly testing cross-chain applications is challenging as sometimes checks depend on the current state of other networks – mocking behaviour is a solution. A more trustworthy approach – but less practical and scalable – is spinning up DLT nodes or leveraging existing test networks, which guarantees a higher level of integration [179]. We highlight that code and testing infrastructure in interoperability projects are ad-hoc designed [181], [246].

6.4.2. Protocol Level. As demonstrated in Table 5, decentralization is an essential requirement for cross-chain solutions – an infrastructure backdoor or a key compromise can be fatal for a solution that presents a single point of failure. Centralized IMs should be less trusted and not manage assets directly. A solution is to insert a higher dependency on the user- or Dapp-specific inputs provided directly to the target chain [82].

Authentication and proof verification mechanisms are crucial components of cross-chain protocols – they must be audited and carefully managed to avoid significant consequences [247]. Access control to contracts with critical functionality must be guaranteed by a studied cross-chain model and architecture and not on ad-hoc practices as it has been until now. Furthermore, we emphasize the importance of architectural decisions such as employing correct incentivization and slashing mechanisms, setting withdrawal limits, and stop-loss procedures. Additionally, our recommendations include the usage of formal frameworks to prove the correctness of protocols (e.g., using UC [248], TLA+ [249], or game theory).

We also highlight a relevant discussion on the contrast between shared and isolated security models [119]. Shared security entails tokens or apps on a given infrastructure adhering to the infrastructure's security requirements, like L2 solutions. In contrast, isolated security allows each app to define its security independently, as seen in user applications built on messaging layers. While isolated security may seem more tailored to specific cases, it raises significant concerns about end-user risk, as users must individually assess the risks associated with each app.

6.4.3. Operational Level. After designing and implementing a cross-chain protocol, the next step is guaranteeing its correct operation. One must protect the system from external malicious parties and maintain the source code updated and bug-free. At the forefront of cross-chain hacks is inadequate key management (USD 1.6B stolen, 55%). Rotating validators' keys or watchers' validation mechanisms can help mitigate the risk of having a centralized single point of failure. Some practices to safeguard private keys are Hardware Security Modules (HSM), Key Management Systems (KMS), or hardware wallets. Setting daily withdrawal limits can also help prevent large-scale losses in the event of an attack. A bug bounty is an attractive option for incentivizing ethical hackers to identify and report vulnerabilities in open-source code (as demonstrated by Table 5 hackers prefer stealing rather than reporting). For example, a white hacker identified an 850M USD vulnerability in the Polygon Plasma bridge resulting in a 2M USD bug bounty [250]. However, open-source software can expose internal mechanisms of a protocol and potential security flaws [251]. Examples include the Wormhole and Thorchain attacks where activity in their public repository (a patch and a comment in the code, respectively) made it easier for attackers to exploit the protocols. Nonetheless, we believe open-source is the way forward to gather efforts from the community. Projects such as LayerZero [118] or Celer [252] have only recently made the code of the *Relayer* and *Validator Network*, respectively, open-source. Developers must strike a balance between promoting transparency and collaboration while also protecting the system. An additional measure that can be implemented is the creation of an insurance fund that is a certain percentage of the TVL. This fund can reimburse users in a security breach or other unforeseen events. Naturally, it is also important to leverage good cybersecurity practices and transversal to the IT sector. Furthermore, we advocate for the assurance that multiple accredited entities verify smart contracts. As an illustration, L2Beat [25] identifies and validates bridge contracts, offering concise project summaries encompassing risk assessments and the most relevant smart contract addresses along with an explanation.

6.5. Future Research Directions

In this section, we lay out future research directions.

6.5.1. Monitoring in Cross-Chain Systems. Given the inherent vulnerabilities in software systems, enhancing the robustness of cross-chain solutions becomes paramount. Initial efforts should focus on the formal verification of cross-chain protocols using an array of tools to augment the likelihood of their correctness. Concurrently, establishing rigorous security and engineering practices for both on-chain and off-chain components is essential. This includes the implementation of automated tests and the meticulous security scrutiny of software dependencies. Proactive prevention can be achieved through the continuous monitoring of all components. Although the Hephaestus framework presents an intriguing direction [32], empirical benchmarks in real-world contexts remain an essential avenue for exploration.

6.5.2. Frameworks for Incident Response in Cross-Chain Contexts. Software platforms interfacing with the internet, especially those governing sensitive tasks like cross-chain bridges, necessitate dedicated cybersecurity oversight. The current research landscape underscores the need for enhanced operational security. Preliminary metrics for detecting bridge discrepancies exist [253], yet manual or automated responses each present their challenges. Mistaken detections, for instance, can result in bridge suspensions, impacting user experience and revenue. To date, comprehensive incident response frameworks for generic cross-chain systems remain largely uncharted, despite some industry-specific endeavors.

6.5.3. Cross-chain Privacy in Heterogeneous Systems. Our findings suggest that cross-chain privacy is predominantly maintained within underlying chains that inherently support privacy-enhancing features. The development and exploration of techniques to ensure unlinkability and anonymity across diverse ledgers remain areas of underexplored research within the scientific community.

6.5.4. Blockchain interoperability design patterns. Design patterns serve as structured frameworks, enabling developers to craft secure solutions with augmented efficacy. Although each interoperability context possesses distinct characteristics, discerning common challenges and pitfalls inherent to specific interoperability solutions can yield invaluable insights. While blockchain design patterns have undergone rigorous scrutiny, a comprehensive examination of design patterns across multifarious blockchain applications remains nascent in the current research landscape, reflecting the evolving nature of this domain.

6.5.5. Data models for blockchain interoperability. Data models are fundamental to interoperability, streamlining complex mappings and varied data formats. Abstract models facilitate a semantic perspective for developers, mirroring the role of SDKs in emphasizing business logic over implementation nuances. Notable strides towards a universal data model are evident through ERC-5164, the ISO model (as adopted by Overledger [254]), SATP Gateways [56], and IBC [61]. However, the path to full standardization remains under exploration, with multiple standards emerging concurrently. The preference for open standards is evident and is crucial for achieving technical interoperability. The importance of this is underscored by initiatives such as BUNGEE [55].

6.5.6. Empirical Investigations. The research landscape reveals a notable gap in empirical studies addressing the detection of theoretical attacks and associated mitigation strategies identified in our analysis. Additionally, there seems to be a scarcity of in-depth examinations focusing on specific IMs. A couple of research trajectories stand out in terms of their pertinence and potential impact. Firstly, the identification of cross-chain Miner Extractable Value (MEV) is becoming increasingly salient due to the rapid expansion of blockchain bridges, coupled with substantial investments to enhance their usability and facilitate the onboarding of newcomers. Secondly, the empirical exploration of oracle manipulation within the cross-chain context [82], [255], [256] presents a promising direction for future investigations.

6.6. Summary:

The importance of comprehensive security in cross-chain operations cannot be understated. Despite the extensive research conducted on cross-chain security, ensuring protection across the entire stack remains imperative. Given the vast attack surface, solely relying on preventive measures, such as continuous monitoring and proactive security, is insufficient. We strongly recommend practitioners bolster their defenses by integrating reactive security measures, including robust incident response frameworks.

Regarding privacy, current research appears to be relatively underexplored. However, as interoperable central bank digital currencies gain traction – evidenced by references like [57], [257] – we foresee a more substantial impetus driving advancements in cross-chain privacy solutions.

Full unlinkability in a permissionless cross-chain scenario is hard to achieve since at least one entity needs to perform the mapping between transactions on both the source and destination chains. We envision that protocols filling this gap will emerge especially with the recent evolution of zero-knowledge technology. In practice, transaction mixers do not yield a high degree of privacy to users due to their naive practices. Solutions circumventing these limitations are necessary. Additionally, existing mixers are being used for malicious activities. Therefore, we highlight the importance of researching how privacy can be guaranteed in regulated and auditable environments.

7. Related Work

Table 7 outlines studies that delve into the security and privacy of blockchain interoperability, juxtaposed with our research. Our study is distinct in the following capacities: 1) We adopt a systematic survey methodology grounded in recognized principles; 2) We fuse both security and privacy dimensions, introducing essential properties requisite for an exhaustive analysis; and 3) We integrate findings from both the gray literature and the industrial sector, essential for comprehensive and rigorous scrutiny. Our research aims to provide developers with pragmatic insights to augment the robustness and privacy of their systems.

TABLE 6. COMPARATIVE ANALYSIS OF THIS PAPER AND OTHER STUDIES FOCUSING ON SECURITY OR PRIVACY OF BLOCKCHAIN INTEROPERABILITY. *All labels apply to systematizations.*

	Security				**Privacy**				**Misc.**	
Ref	P	V	A	M	P	V	A	M	R	IM
[33]	✗	✓^{15}	✓^{1}	✓^{18}	✗	✓^{1}	✗	✓^{1}	46	2
[12]	✗	✓^{29}	✓^{7}	✓^{13}	✗	✓^{1}	✗	✗	–	6
[258]	✗	✗	✗	✗	✓	✗	✗	✓^{4}	–	29
[31]	✗	✓^{11}	✓^{18}	✓^{6}	✗	✗	✗	✗	–	–
this work	✓	✓^{45}	✓^{18}	✓^{93}	✓	✓^{5}	✗	✓^{3}	212	57

✓ – Satisfies criteria
P – Identifies relevant properties
M – Identifies or proposes mitigations
V – Identifies cross-chain vulnerabilities
✗ – Does not satisfy criteria
A – Real-World attacks or leakages
R – Number of relevant references
IM – Number of interoperability mechanisms systematically studied
– Not specified by the authors

7.1. Interoperability and cross-chain rules

During the first five years of research in this field, a significant and influential body of literature has been established. Noteworthy surveys within this realm encompass [3], [7], [10], [47], [259]. The concept of cross-chain rules stands as a foundational pillar for our exploration into the security and privacy intricacies of interoperability. Multiple methodologies have been proposed for the enforcement of these rules. For instance, Ganguly et al. [260] introduce a runtime verification approach that assesses partially synchronous distributed computations, leveraging an SMT-based formula. Within this context, they delineate cross-chain rules through metric temporal logic formulas. Conversely, Hephaestus [32] provides a formal definition of cross-chain rules, positing them as datalog rules upheld by off-chain relayers and smart contracts. This study further lays the groundwork for constructing on-chain use cases adhering to diverse cross-chain logic. Zhang et al. [174] formulate cross-chain rules specifically aimed at detecting discrepancies within the lock-unlock bridge mechanism. While certain studies implicitly address cross-chain rules, others offer more overt definitions, as seen in works discussing oracles [261], [262], bridges [134], and gateways [56].

7.2. Blockchain Security and Attacks

Since the advent of blockchain technology, security has been a focal area of research [263]. An initial endeavor to formalize blockchain security properties was undertaken by Garay et al. [264]. This effort has been followed by numerous refinements, encompassing variable difficulty chains [265], adjustments for participant variability [266], formalizations tailored for proof-of-stake chains [267], specialized considerations for permissioned blockchains [268], among others [269]. Collating this vast knowledge, the literature not only offers surveys centered on the protocol layer [270] but also delves into aspects of implementation [271], [272], network architecture [273], and operational modalities [274], [275]. A significant body of work is dedicated to studying blockchain attacks at various levels: infrastructure/network [263], protocol/application [276]–[278], and operational layers [273].

In the present manuscript, our focus is on the security of IMs, bearing in mind the influence of the security frameworks of the foundational infrastructure.

7.3. Blockchain Privacy

Our investigation into blockchain privacy is informed by seminal research on privacy attributes, specifically anonymity, unlinkability, and confidentiality [279], [280]. The advent of blockchain privacy research coincided with the introduction of the inaugural privacy-centric blockchains [281]. Examples of these pioneering systems include private permissioned networks, such as Fabric [282], privacy-enhanced permissionless blockchains like ZCash and Monero, and applications designed with privacy in mind, such as Tornado Cash. Within the realm of interoperability, a significant portion of privacy research has been concentrated on asset exchanges. Our conceptual model is fundamentally based on the framework proposed by [100]. We have endeavored to broaden this framework to encompass all modes of interoperability.

7.4. A Call for Collaboration

The questions and problems raised here concern the scientific and engineering problems of safely interoperating sets of distributed systems, being safely (and privately) sharing data, or exchanging assets atomically. We believe that blockchain, a powerful technology that brings lots of new possibilities, needs to accommodate today's complex security and privacy landscape. An interdisciplinary approach to the problems referred to in this work has the potential to advance our comprehension of those issues and create a more secure and private ecosystem of ecosystems. We would like to encourage the community to get involved in such efforts. We suggest our open-source repository as an initial point for discussion and exchange of ideas: https://github.com/RafaelAPB/SoKSPBlockchainInterop.

8. Conclusion

This paper systematized relevant security and privacy properties and approaches in blockchain interoperability research. Our study correlates theoretical vulnerabilities and 14 cross-chain bridge hacks, collectively responsible for 94% of the total hacked value up to date. Regarding privacy, our survey reveals a prevalent reliance on zero-knowledge technology. While this method holds promise, it requires extensive additional research before widespread adoption. We collect and propose mitigations for identified vulnerabilities and outline various research pathways, such as reliable monitoring of IMs, frameworks for incident response, the need for empirical studies, and further exploration of cross-chain privacy.

Acknowledgments

We thank Dave Pasirstein and our colleagues in the IETF's SATP working group for fruitful discussions. We appreciate the DPSS group at INESC-ID for suggestions and discussion on this paper. Rafael thanks the Fulbright Association, Thomas Hardjono, and Alex Pentland for the opportunity to research at the MIT Media Lab, where this work was partially developed. We thank Guy Zyskind and colleagues from the MIT Media Lab for fruitful discussions on blockchain privacy and interoperability. Rafael was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UIDB/50021/2020 (INESC-ID) and 2020.06837.BD, and a research scholarship from the Fulbright Association, with the support of FCT. This work was also supported by the BLOCKCHAIN.PT project under reference C632734434-00467077 and by Portuguese national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UIDB/50021/2020 (INESC-ID). Luyao Zhang is supported by the National Science Foundation China on the project entitled "Trust Mechanism Design on Blockchain: An Interdisciplinary Approach of Game Theory, Reinforcement Learning, and Human-AI Interactions (Grant No. 12201266)." Luyao Zhang is also with SciEcon CIC, a not-for-profit (NPO) organization registered in the United Kingdom, aiming to cultivate interdisciplinary research and integrated talents. We thank Kevin Liao for helping us improve the visual quality of this paper.

References

[1] M. Westerkamp and A. Küpper, "SmartSync: Cross-Blockchain Smart Contract Interaction and Synchronization," in *2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, May 2022, pp. 1–9.

[2] P. Gaži, A. Kiayias, and D. Zindros, "Proof-of-stake sidechains," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 139–156.

[3] A. Zamyatin, M. Al-Bassam, D. Zindros, E. Kokoris-Kogias, P. Moreno-Sanchez, A. Kiayias, and W. J. Knottenbelt, "Sok: Communication across distributed ledgers," in *Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science, N. Borisov and C. Diaz, Eds. Berlin, Heidelberg: Springer, 2021, p. 3–36.

[4] P. Wegner, "Interoperability," *ACM Comput. Surv.*, vol. 28, no. 1, p. 285–287, mar 1996. [Online]. Available: https://doi.org/10.1145/234313.234424

[5] D. Engel, M. Herlihy, and Y. Xue, "Failure is (literally) an Option: Atomic Commitment vs Optionality in Decentralized Finance," in *Stabilization, Safety, and Security of Distributed Systems: 23rd International Symposium, SSS 2021, Virtual Event, November 17–20, 2021, Proceedings*. Berlin, Heidelberg: Springer-Verlag, Nov. 2021, pp. 66–77. [Online]. Available: https://doi.org/10.1007/978-3-030-91081-5_5

[6] V. Buterin, "Chain interoperability," *R3 research paper*, vol. 9, pp. 1–25, 2016.

[7] R. Belchior, J. Süßenguth, Q. Feng, T. Hardjono, A. Vasconcelos, and M. Correia, "A Brief History of Blockchain Interoperability," 9 2023. [Online]. Available: https://www.techrxiv.org/articles/preprint/A_Brief_History_of_Blockchain_Interoperability/23418677

[8] G. Wang, "Sok: Exploring blockchains interoperability," *Cryptology ePrint Archive*, 2021.

[9] G. Wang, Q. Wang, and S. Chen, "Exploring Blockchains Interoperability: A Systematic Survey," *ACM Computing Surveys*, p. 3582882, Feb. 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3582882

[10] R. Belchior, A. Vasconcelos, S. Guerreiro, and M. Correia, "A Survey on Blockchain Interoperability: Past, Present, and Future Trends," *ACM Computing Surveys*, vol. 54, no. 8, pp. 168:1–168:41, Oct. 2021. [Online]. Available: https://doi.org/10.1145/3471140

[11] W. Ou, S. Huang, J. Zheng, Q. Zhang, G. Zeng, and W. Han, "An Overview on Cross-chain: Mechanism, Platforms, Challenges and Advances," *Computer Networks*, p. 109378, Sep. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128622004121

[12] L. Duan, Y. Sun, W. Ni, W. Ding, J. Liu, and W. Wang, "Attacks against cross-chain systems and defense approaches: A contemporary survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 8, pp. 1643–1663, 2023.

[13] L. Li, J. Wu, and W. Cui, "A review of blockchain cross-chain technology," *IET Blockchain*, 2023.

[14] H. Mao, T. Nie, H. Sun, D. Shen, and G. Yu, "A Survey on Cross-Chain Technology: Challenges, Development, and Prospect," *IEEE Access*, vol. 11, pp. 45 527–45 546, 2023, conference Name: IEEE Access.

[15] G. Wang, Q. Wang, and S. Chen, "Exploring blockchains interoperability: A systematic survey," *ACM Computing Surveys*, p. 3582882, Feb 2023.

[16] "Top crypto bounty and ransom payments report," 2022. [Online]. Available: https://assets.ctfassets.net/t3wqy70tc3bv/6Tqb2wlVnwdGYeVZX4WDmU/6b0c222b4f680ac80ea801e032894eac/Immunefi_Crypto_Bug_Bounty_and_Ransom_Payments_Report.pdf

[17] "Largest defi exploits." [Online]. Available: https://www.theblock.co/data/decentralized-finance/exploits/largest-defi-exploits

[18] L. Zhou, X. Xiong, J. Ernstberger, S. Chaliasos, Z. Wang, Y. Wang, K. Qin, R. Wattenhofer, D. Song, and A. Gervais, "SoK: Decentralized Finance (DeFi) Attacks," Sep. 2022, arXiv:2208.13035 [cs]. [Online]. Available: http://arxiv.org/abs/2208.13035

[19] "Rekt - leaderboard." [Online]. Available: https://www.rekt.news/

[20] "Rekt - multichain - rekt 2." [Online]. Available: https://rekt.news/multichain-rekt2/

[21] C. Team, "Multichain exploit: Possible hack or rug pull," Jul 2023. [Online]. Available: https://www.chainalysis.com/blog/multichain-exploit-july-2023/

[22] A. [@Allbridge_io], "We are investigating the current situation with the bnb chain pools. the bridge has been temporarily shut down during the investigation. we apologize for the inconvenience." Apr 2023. [Online]. Available: https://twitter.com/Allbridge_io/status/1642341041410908164

[23] S. Reynolds, "Mixin network losses nearly $200m in hack," Sep. 2023. [Online]. Available: https://www.coindesk.com/tech/2023/09/25/mixin-network-losses-nearly-200m-in-hack/

[24] "The chainalysis 2023 crypto crime report," Feb. 2023.

[25] "L2beat – the state of the layer two ecosystem." [Online]. Available: https://l2beat.com/bridges/summary

[26] J. Su, J. Liu, Y. Nan, and Y. Li, "Security Evaluation of Smart Contracts based on Code and Transaction - A Survey," in *2022 International Conference on Service Science (ICSS)*, May 2022, pp. 41–48.

[27] R. Belchior, L. Riley, T. Hardjono, A. Vasconcelos, and M. Correia, "Do you need a distributed ledger technology interoperability solution?" *Distrib. Ledger Technol.*, vol. 2, no. 1, mar 2023. [Online]. Available: https://doi.org/10.1145/3564532

[28] B. Charoenwong and M. Bernardi, "A Decade of Cryptocurrency 'Hacks': 2011 – 2021," Rochester, NY, Oct. 2021. [Online]. Available: https://papers.ssrn.com/abstract=3944435

[29] "Cryptocurrency investigation software - chainalysis reactor." [Online]. Available: https://www.chainalysis.com/chainalysis-reactor/

[30] "Trm labs." [Online]. Available: https://www.trmlabs.com/

[31] Q. Zhao, Y. Wang, B. Yang, K. Shang, M. Sun, H. Wang, Z. Yang, and X. He, "A comprehensive overview of security vulnerability penetration methods in blockchain cross-chain bridges," *Authorea (Authorea)*, Oct 2023. [Online]. Available: https://www.authorea.com/users/674544/articles/672844-a-comprehensive-overview-of-security-vulnerability-penetration-methods-in-blockchain-cross-chain-bridges

[32] R. Belchior, P. Somogyvari, J. Pfannschmid, A. Vasconcelos, and M. Correia, "Hephaestus: Modelling, Analysis, and Performance Evaluation of Cross-Chain Transactions," Sep. 2022.

[33] T. Haugum, B. Hoff, M. Alsadi, and J. Li, "Security and Privacy Challenges in Blockchain Interoperability - A Multivocal Literature Review," in *Proceedings of the International Conference on Evaluation and Assessment in Software Engineering 2022*, ser. EASE '22. New York, NY, USA: Association for Computing Machinery, Jun. 2022, pp. 347–356. [Online]. Available: https://doi.org/10.1145/3530019.3531345

[34] Q. Zhou, H. Huang, Z. Zheng, and J. Bian, "Solutions to Scalability of Blockchain: A Survey," *IEEE Access*, vol. 8, pp. 16 440–16 455, 2020, conference Name: IEEE Access.

[35] V. Buterin, J. Illum, M. Nadler, F. Schär, and A. Soleimani, "Blockchain privacy and regulatory compliance: Towards a practical equilibrium," no. 4563364, Sep 2023. [Online]. Available: https://papers.ssrn.com/abstract=4563364

[36] M. Herlihy, B. Liskov, and L. Shrira, "Cross-chain deals and adversarial commerce," *The VLDB Journal*, vol. 31, no. 6, pp. 1291–1309, Nov. 2022. [Online]. Available: https://doi.org/10.1007/s00778-021-00686-1

[37] H. Tian, K. Xue, X. Luo, S. Li, J. Xu, J. Liu, J. Zhao, and D. S. L. Wei, "Enabling Cross-Chain Transactions: A Decentralized Cryptocurrency Exchange Protocol," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3928–3941, 2021, conference Name: IEEE Transactions on Information Forensics and Security.

[38] M. Herlihy, "Atomic Cross-Chain Swaps," in *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*. Egham United Kingdom: ACM, Jul. 2018, pp. 245–254. [Online]. Available: https://dl.acm.org/doi/10.1145/3212734.3212736

[39] Y. Xue, D. Jin, and M. Herlihy, "Invited Paper: Fault-tolerant and Expressive Cross-Chain Swaps," Nov. 2022, arXiv:2211.00208 [cs]. [Online]. Available: http://arxiv.org/abs/2211.00208

[40] S. Mazumdar, "Towards faster settlement in htlc-based cross-chain atomic swaps," in *2022 IEEE 4th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications (TPS-ISA)*. Los Alamitos, CA, USA: IEEE Computer Society, dec 2022, pp. 295–304. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/TPS-ISA56441.2022.00043

[41] E. Chan, M. Chrobak, and M. Lesani, "Cross-chain Swaps with Preferences," Oct. 2022, arXiv:2210.11791 [cs]. [Online]. Available: http://arxiv.org/abs/2210.11791

[42] S. A. Thyagarajan, G. Malavolta, and P. Moreno-Sánchez, "Universal Atomic Swaps: Secure Exchange of Coins Across All Blockchains," 2021, report Number: 1612. [Online]. Available: https://eprint.iacr.org/2021/1612

[43] Z. Yin, B. Zhang, J. Xu, K. Lu, and K. Ren, "Bool Network: An Open, Distributed, Secure Cross-Chain Notary Platform," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 3465–3478, 2022, conference Name: IEEE Transactions on Information Forensics and Security.

[44] R. Belchior, D. Dimov, Z. Karadjov, J. Pfannschmidt, A. Vasconcelos, and M. Correia, "Harmonia: Securing cross-chain applications using zero-knowledge proofs," 2024, submitted for publication.

[45] W. Li, C. Feng, L. Zhang, H. Xu, B. Cao, and M. A. Imran, "A scalable multi-layer pbft consensus for blockchain," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 5, pp. 1146–1160, 2020.

[46] V. Ramakrishna, "Secure asset transfer protocol (satp) future extensions: Asset and process state queries," IETF 117: Secure Asset Transfer Working Group, Jul. 2023. [Online]. Available: https://datatracker.ietf.org/meeting/117/materials/slides-117-satp-sharing-of-asset-state-and-process-snapshot-views-01

[47] R. Belchior, L. Riley, T. Hardjono, A. Vasconcelos, and M. Correia, "Do You Need a Distributed Ledger Technology Interoperability Solution?" *Distributed Ledger Technologies: Research and Practice*, Sep. 2022, just Accepted. [Online]. Available: https://doi.org/10.1145/3564532

[48] E. Abebe, D. Behl, C. Govindarajan, Y. Hu, D. Karunamoorthy, P. Novotny, V. Pandit, V. Ramakrishna, and C. Vecchiola, "Enabling enterprise blockchain interoperability with trusted data transfer (industry track)," in *Proceedings of the 20th International Middleware Conference Industrial Track*. Davis CA USA: ACM, Dec 2019, p. 29–35. [Online]. Available: https://dl.acm.org/doi/10.1145/3366626.3368129

[49] G. Wang, Q. Wang, and S. Chen, "Exploring blockchains interoperability: A systematic survey," *ACM Computing Surveys*, 2023.

[50] P. Frauenthaler, M. Sigwart, C. Spanring, M. Sober, and S. Schulte, "ETH Relay: A Cost-efficient Relay for Ethereum-based Blockchains," in *2020 IEEE International Conference on Blockchain (Blockchain)*, Nov. 2020, pp. 204–213.

[51] O. Ciobotaru, F. Shirazi, A. Stewart, and S. Vasilyev, "Accountable light client systems for pos blockchains," Cryptology ePrint Archive, Paper 2022/1205, 2022, https://eprint.iacr.org/2022/1205. [Online]. Available: https://eprint.iacr.org/2022/1205

[52] T. Xie, J. Zhang, Z. Cheng, F. Zhang, Y. Zhang, Y. Jia, D. Boneh, and D. Song, "zkBridge: Trustless Cross-chain Bridges Made Practical," Oct. 2022, arXiv:2210.00264 [cs]. [Online]. Available: http://arxiv.org/abs/2210.00264

[53] M. Westerkamp and J. Eberhardt, "zkRelay: Facilitating Sidechains using zkSNARK-based Chain-Relays," in *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, Sep. 2020, pp. 378–386.

[54] [Online]. Available: https://stargate.finance/

[55] R. Belchior, L. Torres, J. Pfannschmid, A. Vasconcelos, and M. Correia, "Can We Share the Same Perspective? Blockchain Interoperability with Views," Oct. 2022.

[56] M. Hargreaves, T. Hardjono, and R. Belchior, *Secure Asset Transfer Protocol (SATP)*, Jul 2023, no. draft-ietf-satp-core-02. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-satp-core

[57] A. Augusto, R. Belchior, I. Kocsis, L. Gönczy, A. Vasconcelos, and M. Correia, "Cbdc bridging between hyperledger fabric and permissioned evm-based blockchains," in *2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 2023, pp. 1–9.

[58] A. Zamyatin, D. Harz, J. Lind, P. Panayiotou, A. Gervais, and W. Knottenbelt, "XCLAIM: Trustless, Interoperable, Cryptocurrency-Backed Assets," in *2019 IEEE Symposium on Security and Privacy (SP)*, May 2019, pp. 193–210, iSSN: 2375-1207.

[59] A. Sapirshtein, Y. Sompolinsky, and A. Zohar, "Optimal selfish mining strategies in bitcoin," in *Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science, J. Grossklags and B. Preneel, Eds. Berlin, Heidelberg: Springer, 2017, p. 515–532.

[60] T. Gauthier, S. Dan, M. Hadji, A. Del Pozzo, and Y. Amoussou-Guenou, "Topos: A Secure, Trustless, and Decentralized Interoperability Protocol," Feb. 2023, arXiv:2206.03481 [cs]. [Online]. Available: http://arxiv.org/abs/2206.03481

[61] J. Kwon and E. Buchman, "Cosmos whitepaper," *A Netw. Distrib. Ledgers*, vol. 27, 2019.

[62] G. Wood, "Polkadot: Vision for a heterogeneous multi-chain framework," *White paper*, vol. 21, no. 2327, p. 4662, 2016.

[63] "Ethereum whitepaper." [Online]. Available: https://ethereum.org

[64] E. Abebe, P. Robinson, A. Chand, M. Murdock, and D. Hyland-Wood, "Crosschain Risk Framework." [Online]. Available: https://crosschainriskframework.github.io/

[65] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, Jan. 2004, conference Name: IEEE Transactions on Dependable and Secure Computing.

[66] E. J. Scheid, T. Hegnauer, B. Rodrigues, and B. Stiller, "Bifröst: a modular blockchain interoperability api," in *2019 IEEE 44th Conference on Local Computer Networks (LCN)*, Oct 2019, p. 332–339.

[67] S. Ghaemi, S. Rouhani, R. Belchior, R. S. Cruz, H. Khazaei, and P. Musilek, "A pub-sub architecture to promote blockchain interoperability," 2021.

[68] Y. Tao, B. Li, and B. Li, "On atomicity and confidentiality across blockchains under failures," *IEEE Transactions on Knowledge and Data Engineering*, p. 1–14, 2023.

[69] L. Vishwakarma, A. Kumar, and D. Das, "Crossledger: A pioneer cross-chain asset transfer protocol," in *2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, May 2023, p. 568–578.

[70] Y. Zhang, S. Hu, Q. Wang, B. Qin, Q. Wu, and W. Shi, "PXCrypto: A Regulated Privacy-Preserving Cross-Chain Transaction Scheme," in *Algorithms and Architectures for Parallel Processing*, ser. Lecture Notes in Computer Science, W. Meng, R. Lu, G. Min, and J. Vaidya, Eds. Cham: Springer Nature Switzerland, 2023, pp. 170–191.

[71] R. Belchior, A. Vasconcelos, M. Correia, and T. Hardjono, "Hermes: Fault-tolerant middleware for blockchain interoperability," *Future Generation Computer Systems*, vol. 129, pp. 236–251, Apr. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X21004337

[72] D. Patel, H. Anand, and S. Chakraborty, "CrossTrustchain: Cross-Chain Interoperability using Multivariate Trust Models," in *2023 15th International Conference on COMmunication Systems & NETworkS (COMSNETS)*, Jan. 2023, pp. 129–134, iSSN: 2155-2509.

[73] S. Zhang, T. Xie, K. Gai, and L. Xu, "ARC: An Asynchronous Consensus and Relay Chain-based Cross-chain Solution to Consortium Blockchain," in *2022 IEEE 9th International Conference on Cyber Security and Cloud Computing (CSCloud)/2022 IEEE 8th International Conference on Edge Computing and Scalable Cloud (EdgeCom)*, Jun. 2022, pp. 86–92, iSSN: 2693-8928.

[74] O. Shlomovits and O. Leiba, "JugglingSwap: Scriptless Atomic Cross-Chain Swaps," Jul. 2020, arXiv:2007.14423 [cs]. [Online]. Available: http://arxiv.org/abs/2007.14423

[75] B. Dai, S. Jiang, M. Zhu, M. Lu, D. Li, and C. Li, "Research and implementation of cross-chain transaction model based on improved hash-locking," in *Blockchain and Trustworthy Systems*, ser. Communications in Computer and Information Science, Z. Zheng, H.-N. Dai, X. Fu, and B. Chen, Eds. Singapore: Springer, 2020, p. 218–230.

[76] Y. Lan, J. Gao, Y. Li, K. Wang, Y. Zhu, and Z. Chen, "Trustcross: Enabling confidential interoperability across blockchains using trusted hardware," in *2021 4th International Conference on Blockchain Technology and Applications*. Xi'an China: ACM, Dec 2021, p. 17–23. [Online]. Available: https://dl.acm.org/doi/10.1145/3510487.3510491

[77] M. Li, J. Weng, Y. Li, Y. Wu, J. Weng, D. Li, G. Xu, and R. Deng, "IvyCross: A Privacy-Preserving and Concurrency Control Framework for Blockchain Interoperability," 2021, report Number: 1244. [Online]. Available: https://eprint.iacr.org/2021/1244

[78] G. Wang and M. Nixon, "InterTrust: Towards an Efficient Blockchain Interoperability Architecture with Trusted Services," in *2021 IEEE International Conference on Blockchain (Blockchain)*, Dec. 2021, pp. 150–159.

[79] I. Bentov, Y. Ji, F. Zhang, L. Breidenbach, P. Daian, and A. Juels, "Tesseract: Real-Time Cryptocurrency Exchange Using Trusted Hardware," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '19. New York, NY, USA: Association for Computing Machinery, Nov. 2019, pp. 1521–1538. [Online]. Available: https://doi.org/10.1145/3319535.3363221

[80] V. Zakhary, D. Agrawal, and A. El Abbadi, "Atomic commitment across blockchains," *Proceedings of the VLDB Endowment*, vol. 13, no. 9, p. 1319–1331, May 2020.

[81] Y. Li, J. Weng, M. Li, W. Wu, J. Weng, J.-N. Liu, and S. Hu, "ZeroCross: A sidechain-based privacy-preserving Cross-chain solution for Monero," *Journal of Parallel and Distributed Computing*, vol. 169, pp. 301–316, Nov. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0743731522001733

[82] M. D. Montiel, R. Guerraoui, and P.-L. Roman, "SurferMonkey: A Decentralized Anonymous Blockchain Intercommunication System via Zero Knowledge Proofs," Oct. 2022, arXiv:2210.13242 [cs]. [Online]. Available: http://arxiv.org/abs/2210.13242

[83] W. Liu, Z. Wan, J. Shao, and Y. Yu, "HyperMaze: Towards Privacy-Preserving and Scalable Permissioned Blockchain," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 1, pp. 360–376, Jan. 2023, conference Name: IEEE Transactions on Dependable and Secure Computing.

[84] X. Pang, N. Kong, and Z. Chen, "AbitBridge: A cross-chain protocol based on main-sub-chain architecture," in *2022 IEEE 5th International Conference on Information Systems and Computer Aided Education (ICISCAE)*, Sep. 2022, pp. 99–104, iSSN: 2770-663X.

[85] Z. Liu, Y. Xiang, J. Shi, P. Gao, H. Wang, X. Xiao, B. Wen, and Y.-C. Hu, "HyperService: Interoperability and Programmability Across Heterogeneous Blockchains," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. London United Kingdom: ACM, Nov. 2019, pp. 549–566. [Online]. Available: https://dl.acm.org/doi/10.1145/3319535.3355503

[86] B. C. Ghosh, T. Bhartia, S. K. Addya, and S. Chakraborty, "Leveraging Public-Private Blockchain Interoperability for Closed Consortium Interfacing," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, May 2021, pp. 1–10, iSSN: 2641-9874.

[87] Y. Sun, L. Yi, L. Duan, and W. Wang, "A Decentralized Cross-Chain Service Protocol based on Notary Schemes and Hash-Locking," in *2022 IEEE International Conference on Services Computing (SCC)*, Jul. 2022, pp. 152–157, iSSN: 2474-2473.

[88] M. Sober, G. Scaffino, C. Spanring, and S. Schulte, "A Voting-Based Blockchain Interoperability Oracle," Nov. 2021, arXiv:2111.10091 [cs]. [Online]. Available: http://arxiv.org/abs/2111.10091

[89] Y. Yang, F. Bai, Z. Yu, T. Shen, Y. Liu, and B. Gong, "An Anonymous and Supervisory Cross-Chain Privacy Protection Protocol for Zero-Trust IoT Application," *ACM Transactions on Sensor Networks*, p. 3583073, Mar. 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3583073

[90] F. Barbàra and C. Schifanella, "BxTB: cross-chain exchanges of bitcoins for all Bitcoin wrapped tokens," in *2022 Fourth International Conference on Blockchain Computing and Applications (BCCA)*, Sep. 2022, pp. 143–150.

[91] M. Westerkamp and M. Diez, "Verilay: A Verifiable Proof of Stake Chain Relay," in *2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, May 2022, pp. 1–9, arXiv:2201.08697 [cs]. [Online]. Available: http://arxiv.org/abs/2201.08697

[92] A. Sanchez, A. Stewart, and F. Shirazi, "Bridging Sapling: Private Cross-Chain Transfers," in *2022 IEEE Crosschain Workshop (ICBC-CROSS)*, May 2022, pp. 1–9.

[93] D. Stone, "Trustless, privacy-preserving blockchain bridges," Feb. 2021, arXiv:2102.04660 [cs]. [Online]. Available: http://arxiv.org/abs/2102.04660

[94] A. Li, G. D'Angelo, J. Tang, F. Fang, and B. Gong, "An auditable confidentiality protocol for blockchain transactions," Cryptology ePrint Archive, Paper 2022/1672, 2022. [Online]. Available: https://eprint.iacr.org/2022/1672

[95] Y. Xue and M. Herlihy, "Hedging Against Sore Loser Attacks in Cross-Chain Transactions," in *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*, ser. PODC'21. New York, NY, USA: Association for Computing Machinery, Jul. 2021, pp. 155–164. [Online]. Available: https://doi.org/10.1145/3465084.3467904

[96] X. Zhang, J. Chen, Y. Zhou, and S. Jiang, "Privacy-Preserving Cross-Chain Payment Scheme for Blockchain-Enabled Energy Trading," in *2021 IEEE/CIC International Conference on Communications in China (ICCC)*, Jul. 2021, pp. 109–114, iSSN: 2377-8644.

[97] D. Ding, B. Long, F. Zhuo, Z. Li, H. Zhang, C. Tian, and Y. Sun, "Lilac: Parallelizing Atomic Cross-Chain Swaps," in *2022 IEEE Symposium on Computers and Communications (ISCC)*, Jun. 2022, pp. 1–8, iSSN: 2642-7389.

[98] T. Bugnet and A. Zamyatin, "XCC: Theft-Resilient and Collateral-Optimized Cryptocurrency-Backed Assets," 2022, report Number: 113. [Online]. Available: https://eprint.iacr.org/2022/113

[99] I. Tsabary, M. Yechieli, A. Manuskin, and I. Eyal, "MAD-HTLC: Because HTLC is Crazy-Cheap to Attack," in *2021 IEEE Symposium on Security and Privacy (SP)*, May 2021, pp. 1230–1248, iSSN: 2375-1207.

[100] A. Deshpande and M. Herlihy, "Privacy-Preserving Cross-Chain Atomic Swaps," in *Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science, M. Bernhard, A. Bracciali, L. J. Camp, S. Matsuo, A. Maurushat, P. B. Rønne, and M. Sala, Eds. Cham: Springer International Publishing, 2020, pp. 540–549.

[101] J. Cai, Y. Zhou, T. Hu, and B. Li, "PTLC: Protect the Identity Privacy during Cross-Chain Asset Transaction More Effectively," in *2022 IEEE 22nd International Conference on Software Quality, Reliability, and Security Companion (QRS-C)*, Dec. 2022, pp. 70–78, iSSN: 2693-9371.

[102] J. Kirsten and H. Davarpanah, “Anonymous Atomic Swaps Using Homomorphic Hashing,” Rochester, NY, Aug. 2018. [Online]. Available: https://papers.ssrn.com/abstract=3235955

[103] K. Narayanam, V. Ramakrishna, D. Vinayagamurthy, and S. Nishad, “Atomic cross-chain exchanges of shared assets,” Sep. 2022, arXiv:2202.12855 [cs]. [Online]. Available: http://arxiv.org/abs/2202.12855

[104] R. Li, Y. Xie, Z. Ning, C. Zhang, and L. Wei, “Privacy-Preserving Decentralized Cryptocurrency Exchange without Price Manipulation,” in *2022 IEEE/CIC International Conference on Communications in China (ICCC)*, Aug. 2022, pp. 274–279, iSSN: 2377-8644.

[105] L. Hanzlik, J. Loss, S. A. Thyagarajan, and B. Wagner, “Sweep-uc: Swapping coins privately,” Cryptology ePrint Archive, Paper 2022/1605, 2022. [Online]. Available: https://eprint.iacr.org/2022/1605

[106] Y. Manevich and A. Akavia, “Cross Chain Atomic Swaps in the Absence of Time via Attribute Verifiable Timed Commitments,” in *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, Jun. 2022, pp. 606–625.

[107] T. Hardjono and N. Smith, “Decentralized Trusted Computing Base for Blockchain Infrastructure Security,” *Frontiers in Blockchain*, vol. 2, 2019. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fbloc.2019.00024

[108] L. T. Thibault, T. Sarry, and A. S. Hafid, “Blockchain Scaling Using Rollups: A Comprehensive Survey,” *IEEE Access*, vol. 10, pp. 93 039–93 054, 2022, conference Name: IEEE Access.

[109] “Crypto giant ftx collapses into bankruptcy,” *BBC News*, Nov 2022. [Online]. Available: https://www.bbc.com/news/business-63601213

[110] “Report on Crypto Exchange Hacks.” [Online]. Available: https://cointelegraph.com/magazine/crypto-exchange-hacks/

[111] “Coinex faces a major security breach with $27 million estimated loss – cryptopolitan.” [Online]. Available: https://www.cryptopolitan.com/coinex-faces-a-major-security-breach/

[112] “Announcement | Binance Security Breach Update.” [Online]. Available: https://www.binance.com/en/support/announcement/binance-security-breach-update-360028031711

[113] M. Sacramento, “Crypto Exchange Bitfinex Bounces Back after a DDoS Attack,” Jun 2018. [Online]. Available: https://www.ccn.com/crypto-exchange-bitfinex-bounces-back-after-a-ddos-attack/

[114] M. Sabt, M. Achemlal, and A. Bouabdallah, “Trusted execution environment: What it is, and what it is not,” in *2015 IEEE Trustcom/BigDataSE/ISPA*, vol. 1, 2015, pp. 57–64.

[115] A. Labs, “Avalanche bridge.” [Online]. Available: https://core.app/bridge/

[116] “Why trusted execution environments will be integral to proof-of-stake blockchains,” Jun 2022. [Online]. Available: https://venturebeat.com/datadecisionmakers/why-trusted-execution-environments-will-be-integral-to-proof-of-stake-blockchains/

[117] “What is the role of the avalanche bridge nodes?” [Online]. Available: https://support.avax.network/en/articles/5462271-what-is-the-role-of-the-avalanche-bridge-nodes

[118] R. Zarick, B. Pellegrino, and C. Banister, “Layerzero: Trustless omnichain interoperability protocol,” 2021.

[119] K. Urbański, “Circumventing layer zero,” Jan 2023. [Online]. Available: https://medium.com/l2beat/circumventing-layer-zero-5e9f652a5d3e

[120] P. Jauernig, A.-R. Sadeghi, and E. Stapf, “Trusted execution environments: Properties, applications, and challenges,” *IEEE Security & Privacy*, vol. 18, no. 2, pp. 56–60, 2020.

[121] “Axelar Network: Connecting Applications with Blockchain Ecosystems,” 2021. [Online]. Available: https://axelar.network/axelar_whitepaper.pdf

[122] “The value layer of the internet.” [Online]. Available: https://polygon.technology/

[123] “Portal token bridge.” [Online]. Available: https://portalbridge.com

[124] “Bridge assesment report – uniswap foundation.” [Online]. Available: https://uniswap.notion.site/Bridge-Assessment-Report-0c8477afadce425abac9c0bd175ca382

[125] [Online]. Available: https://docs.axelar.dev/learn/security

[126] K.-H. Yeh, G.-Y. Yang, C. Butpheng, L.-F. Lee, and Y.-H. Liu, “A Secure Interoperability Management Scheme for Cross-Blockchain Transactions,” *Symmetry*, vol. 14, no. 12, p. 2473, Dec. 2022, number: 12 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/2073-8994/14/12/2473

[127] “Wanchain – we are all connected.” [Online]. Available: https://docs.wanchain.org/get-started/introduction

[128] “Btc relay.” [Online]. Available: http://btcrelay.org/

[129] “Minimal light client.” [Online]. Available: https://github.com/ethereum/annotated-spec/blob/master/altair/sync-protocol.md

[130] “zkrouter,” Nov. 2022. [Online]. Available: https://drive.google.com/file/d/1ibuHChcYcYCN6JelRAQPnM4rkaB9EgAM

[131] “zksync — accelerating the mass adoption of crypto for personal sovereignty.” [Online]. Available: https://zksync.io/

[132] “Scroll - native zkevm layer 2 for ethereum.” [Online]. Available: https://scroll.io/

[133] “Taiko.” [Online]. Available: https://taiko.xyz/

[134] Bhuptani, Arjun, “Optimistic Bridges: A New Paradigm for Crosschain Communication,” 2022, available online: https://blog.connext.network/optimistic-bridges-fb800dc7b0e0, last accessed on 2023-05-21. [Online]. Available: https://blog.connext.network/optimistic-bridges-fb800dc7b0e0

[135] “Optimism.” [Online]. Available: https://www.optimism.io/

[136] M. Zecchini, M. Sober, S. Schulte, and A. Vitaletti, “Building a cross-chain identity: A self-sovereign identity-based framework,” in *2023 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*, 2023, pp. 149–156.

[137] R. Han, H. Lin, and J. Yu, “On the optionality and fairness of atomic swaps,” in *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, ser. AFT ’19. New York, NY, USA: Association for Computing Machinery, 2019, p. 62–75. [Online]. Available: https://doi.org/10.1145/3318041.3355460

[138] J. Xu, D. Ackerer, and A. Dubovitskaya, “A Game-Theoretic Analysis of Cross-Chain Atomic Swaps with HTLCs,” in *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, Jul. 2021, pp. 584–594, iSSN: 2575-8411.

[139] D. Boneh and M. Naor, “Timed commitments,” in *Advances in Cryptology — CRYPTO 2000*, ser. Lecture Notes in Computer Science, M. Bellare, Ed. Berlin, Heidelberg: Springer, 2000, p. 236–254.

[140] S. A. K. Thyagarajan, A. Bhat, G. Malavolta, N. Döttling, A. Kate, and D. Schröder, “Verifiable timed signatures made practical,” in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS ’20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1733–1750. [Online]. Available: https://doi.org/10.1145/3372297.3417263

[141] H. Yousaf, G. Kappos, and S. Meiklejohn, “Tracing transactions across cryptocurrency ledgers,” in *Proceedings of the 28th USENIX Conference on Security Symposium*, ser. SEC’19. USA: USENIX Association, Aug. 2019, pp. 837–850.

[142] Q. Wang, B. Qin, J. Hu, and F. Xiao, “Preserving transaction privacy in bitcoin,” *Future Generation Computer Systems*, vol. 107, pp. 793–804, Jun. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X17318393

[143] H. Xie, S. Fei, Z. Yan, and Y. Xiao, “SofitMix: A Secure Offchain-Supported Bitcoin-Compatible Mixing Protocol,” *IEEE Transactions on Dependable and Secure Computing*, pp. 1–15, 2022, conference Name: IEEE Transactions on Dependable and Secure Computing.

[144] F. A. Hayek, M. Koscina, P. Lafourcade, and C. Olivier-Anclin, “Generic Privacy Preserving Private Permissioned Blockchains,” in *The 38th ACM/SIGAPP Symposium On Applied Computing*, Tallinn, Estonia, Mar. 2023. [Online]. Available: https://hal.uca.fr/hal-03906880

[145] Z. Wang, S. Chaliasos, K. Qin, L. Zhou, L. Gao, P. Berrang, B. Livshits, and A. Gervais, “On How Zero-Knowledge Proof Blockchain Mixers Improve, and Worsen User Privacy,” Jan. 2022, arXiv:2201.09035 [cs]. [Online]. Available: http://arxiv.org/abs/2201.09035

[146] “Tornado cash.” [Online]. Available: https://github.com/tornadocash

[147] G. Almashaqbeh and R. Solomon, "Sok: Privacy-preserving computing in the blockchain era," in *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, Jun 2022, p. 124–139.

[148] Z. Guan, Z. Wan, Y. Yang, Y. Zhou, and B. Huang, "BlockMaze: An Efficient Privacy-Preserving Account-Model Blockchain Based on zk-SNARKs," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 3, pp. 1446–1463, May 2022, conference Name: IEEE Transactions on Dependable and Secure Computing.

[149] Q. Feng, D. He, S. Zeadally, M. K. Khan, and N. Kumar, "A survey on privacy protection in blockchain system," *Journal of Network and Computer Applications*, vol. 126, p. 45–58, Jan 2019.

[150] ——, "A survey on privacy protection in blockchain system," *Journal of Network and Computer Applications*, vol. 126, pp. 45–58, Jan. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804518303485

[151] C. McMenamin, "Sok: Cross-domain mev," *arXiv preprint arXiv:2308.04159*, 2023.

[152] Z. Bao, W. Shi, S. Kumari, Z.-y. Kong, and C.-M. Chen, "Lockmix: a secure and privacy-preserving mix service for Bitcoin anonymity," *International Journal of Information Security*, vol. 19, no. 3, pp. 311–321, Jun. 2020. [Online]. Available: https://doi.org/10.1007/s10207-019-00459-6

[153] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, Feb. 1981. [Online]. Available: https://dl.acm.org/doi/10.1145/358549.358563

[154] J. Swihart, B. Winston, and S. Bowe, "Zcash counterfeiting vulnerability successfully remediated," *Retrieved November*, vol. 20, p. 2019, 2019.

[155] "U.s. treasury sanctions notorious virtual currency mixer tornado cash," Aug. 2023. [Online]. Available: https://home.treasury.gov/news/press-releases/jy0916

[156] Z. Wang, S. Chaliasos, K. Qin, L. Zhou, L. Gao, P. Berrang, B. Livshits, and A. Gervais, "On how zero-knowledge proof blockchain mixers improve, and worsen user privacy," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 2022–2032.

[157] D. Chaum, "Blind signature system," in *Advances in Cryptology: Proceedings of Crypto 83*. Springer, 1983, pp. 153–153.

[158] N. Alsalami and B. Zhang, "SoK: A Systematic Study of Anonymity in Cryptocurrencies," in *2019 IEEE Conference on Dependable and Secure Computing (DSC)*, Nov. 2019, pp. 1–9.

[159] L. Wang, G. Zhang, and C. Ma, "A survey of ring signature," *Frontiers of Electrical and Electronic Engineering in China*, vol. 3, no. 1, p. 10–19, Jan. 2008.

[160] "The monero project." [Online]. Available: https://www.getmonero.org/index.html

[161] J. Lv and X. Wang, "Verifiable ring signature," in *Proc. of DMS 2003-The 9th International Conference on Distribted Multimedia Systems*, 2003, pp. 663–667.

[162] "Polygon." [Online]. Available: https://wiki.polygon.technology/

[163] "Optmism." [Online]. Available: https://community.optimism.io/docs/developers/bridge/basics.html

[164] "Arbitrum." [Online]. Available: https://docs.arbitrum.io/devs-how-tos/bridge-tokens/how-to-bridge-tokens-overview

[165] "Ronin." [Online]. Available: https://docs.roninchain.com/docs/basics/dapps/ronin-bridge

[166] "zksync - bridging." [Online]. Available: https://era.zksync.io/docs/reference/concepts/bridging-asset.html

[167] "Connext." [Online]. Available: https://docs.connext.network/concepts/readme

[168] L. Zhang, X. Ma, and Y. Liu, "Sok: Blockchain decentralization," *arXiv preprint arXiv:2205.04256*, 2022.

[169] R. Belchior, S. Scuri, I. Mihaiu, N. Nunes, and T. Hardjono, "Towards a Common Standard Framework for Blockchain Interoperability - A Position Paper," 10 2023. [Online]. Available: https://www.techrxiv.org/articles/preprint/A_Framework_to_Evaluate_Blockchain_Interoperability_Solutions/17093039

[170] M. Wu, W. McTighe, K. Wang, I. A. Seres, N. Bax, M. Puebla, M. Mendez, F. Carrone, T. D. Mattey, H. O. Demaestri, M. Nicolini, and P. Fontana, "Tutela: An open-source tool for assessing user-privacy on ethereum and tornado cash," 2022.

[171] "Celestia." [Online]. Available: https://celestia.org/

[172] "Arbitrum audit." [Online]. Available: https://github.com/ArbitrumFoundation/governance/blob/main/audits/trail_of_bits_governance_report_1_6_2023.pdf

[173] S.-S. Lee, A. Murashkin, M. Derka, and J. Gorzny, "SoK: Not Quite Water Under the Bridge: Review of Cross-Chain Bridge Hacks," Oct. 2022, arXiv:2210.16209 [cs]. [Online]. Available: http://arxiv.org/abs/2210.16209

[174] J. Zhang, J. Gao, Y. Li, Z. Chen, Z. Guan, and Z. Chen, "Xscope: Hunting for Cross-Chain Bridge Attacks," Aug. 2022, arXiv:2208.07119 [cs]. [Online]. Available: http://arxiv.org/abs/2208.07119

[175] "Axie infinity bridge audit." [Online]. Available: https://docs.roninchain.com/assets/files/CertiK-Audit-for-Axie-Infinity---Audit-v8-1bfcb82b195442bf34a28ed2fdbde6c5.pdf

[176] Z. Lv, D. Wu, W. Yang, and L. Duan, "Attack and protection schemes on fabric isomorphic crosschain systems," *International Journal of Distributed Sensor Networks*, vol. 18, no. 1, p. 15501477211059945, Jan. 2022, publisher: SAGE Publications. [Online]. Available: https://doi.org/10.1177/15501477211059945

[177] Jun 2022. [Online]. Available: https://aurora.dev/blog/aurora-mitigates-its-inflation-vulnerability

[178] "Starknet dai bridge audit." [Online]. Available: https://chainsecurity.com/wp-content/uploads/2021/12/ChainSecurity_MakerDAO_StarkNet-DAI-Bridge_audit.pdf

[179] T. Eizinger, P. Hoenisch, and L. S. del Pino, "Open problems in cross-chain protocols," Jan. 2021, arXiv:2101.12412 [cs]. [Online]. Available: http://arxiv.org/abs/2101.12412

[180] "Message traps in the arbitrum bridge," 2022. [Online]. Available: https://www.notonlyowner.com/research/message-traps-in-the-arbitrum-bridge

[181] "Wormhole audit." [Online]. Available: https://github.com/trailofbits/publications/blob/master/reviews/2023-03-wormhole-securityreview.pdf

[182] "Circle audit." [Online]. Available: https://chainsecurity.com/wp-content/uploads/2023/04/Circle-Smart-Contract-Audit-_-Cross-Chain-Transfer-Protocol-CCTP-_-EVM-Bridge-_-ChainSecurity.pdf

[183] A. S. Aiyer, L. Alvisi, A. Clement, M. Dahlin, J.-P. Martin, and C. Porth, "Bar fault tolerance for cooperative services," in *Proceedings of the twentieth ACM symposium on Operating systems principles*, ser. SOSP '05. New York, NY, USA: Association for Computing Machinery, Oct 2005, p. 45–58. [Online]. Available: https://doi.org/10.1145/1095810.1095816

[184] A. Judmayer, N. Stifter, A. Zamyatin, I. Tsabary, I. Eyal, P. Gazi, S. Meiklejohn, and E. Weippl, "Pay to win: Cheap, crowdfundable, cross-chain algorithmic incentive manipulation attacks on pow cryptocurrencies," Cryptology ePrint Archive, Paper 2019/775, 2019. [Online]. Available: https://eprint.iacr.org/2019/775

[185] F. Winzer, B. Herd, and S. Faust, "Temporary censorship attacks in the presence of rational miners," in *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, Jun 2019, p. 357–366.

[186] A. Pupyshev, D. Gubanov, E. Dzhafarov, I. Sapranidi, I. Kardanov, V. Zhuravlev, S. Khalilov, M. Jansen, S. Laureyssens, I. Pavlov, and S. Ivanov, "Gravity: a blockchain-agnostic cross-chain communication and data oracles protocol," Aug. 2020, arXiv:2007.00966 [cs]. [Online]. Available: http://arxiv.org/abs/2007.00966

[187] L. Gudgeon, P. Moreno-Sanchez, S. Roos, P. McCorry, and A. Gervais, "SoK: Layer-Two Blockchain Protocols," in *Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science, J. Bonneau and N. Heninger, Eds. Cham: Springer International Publishing, 2020, pp. 201–226.

[188] W. Foundation, "wormhole/SECURITY.md at main · wormhole-foundation/wormhole — github.com," https://github.com/wormhole-foundation/wormhole/blob/main/SECURITY.md, [Accessed 07-Jul-2023].

[189] B. Mazorra, M. Reynolds, and V. Daza, “Price of mev: towards a game theoretical approach to mev,” in *Proceedings of the 2022 ACM CCS Workshop on Decentralized Finance and Security*, 2022, pp. 15–22.

[190] P. McCorry, C. Buckland, B. Yee, and D. Song, “Sok: Validating bridges as a scaling solution for blockchains,” Cryptology ePrint Archive, Paper 2021/1589, 2021. [Online]. Available: https://eprint.iacr.org/2021/1589

[191] L. Luu, J. Teutsch, R. Kulkarni, and P. Saxena, “Demystifying incentives in the consensus computer,” in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS ’15. New York, NY, USA: Association for Computing Machinery, Oct 2015, p. 706–719. [Online]. Available: https://doi.org/10.1145/2810103.2813659

[192] S. M. Werner, D. Perez, L. Gudgeon, A. Klages-Mundt, D. Harz, and W. J. Knottenbelt, “Sok: Decentralized finance (defi),” *arXiv preprint arXiv:2101.08778*, 2021.

[193] T. Mackinga, T. Nadahalli, and R. Wattenhofer, “Twap oracle attacks: Easier done than said?” in *2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2022, pp. 1–8.

[194] K. Tjiam, R. Wang, H. Chen, and K. Liang, “Your smart contracts are not secure: investigating arbitrageurs and oracle manipulators in ethereum,” in *Proceedings of the 3rd Workshop on Cyber-Security Arms Race*, 2021, pp. 25–35.

[195] S. Eskandari, M. Salehi, W. C. Gu, and J. Clark, “Sok: Oracles from the ground truth to market manipulation,” in *Proceedings of the 3rd ACM Conference on Advances in Financial Technologies*, 2021, pp. 127–141.

[196] Q. Finance, “Protocol exploit report,” Jan 2022. [Online]. Available: https://medium.com/@QubitFin/protocol-exploit-report-305c34540fa3

[197] “Rekt - meter.” [Online]. Available: https://rekt.news/meter-rekt/

[198] “Rekt - nomad bridge.” [Online]. Available: https://rekt.news/nomad-rekt/

[199] THORChain, “Eth parsing error and exploit,” Jun 2021. [Online]. Available: https://medium.com/thorchain/eth-parsing-error-and-exploit-3b343aa6466f

[200] M. Gupta, “Poly network hack analysis - largest crypto hack,” Aug 2021. [Online]. Available: https://mudit.blog/poly-network-largest-crypto-hack/

[201] “Rekt - bnb bridge.” [Online]. Available: https://www.rekt.news/bnb-bridge-rekt/

[202] C. [@CelerNetwork], “(1/n)a dns cache poisoning attack on cbridge’s frontend ui approx...” Aug 2022. [Online]. Available: https://twitter.com/CelerNetwork/status/1560123830844411904

[203] “Multichain contract vulnerability post mortem | by multichain (previously anyswap) | medium.” [Online]. Available: https://medium.com/multichainorg/multichain-contract-vulnerability-post-mortem-d37bfab237c8

[204] “Wormhole audit.” [Online]. Available: https://github.com/wormhole-foundation/wormhole-audits/blob/main/Wormhole_Audit_Report_TrailOfBits_2022-09.pdf

[205] “Wormhole audit.” [Online]. Available: https://github.com/wormhole-foundation/wormhole-audits/blob/main/2023-03-08_CertiK_Wormhole_EVM.pdf

[206] “Polygon pos audit.” [Online]. Available: https://chainsecurity.com/wp-content/uploads/2023/04/Polygon_PoS_Portal_-Smart-Contract-Audit_ChainSecurity.pdf

[207] “zksync dai bridge audit.” [Online]. Available: https://chainsecurity.com/wp-content/uploads/2023/08/ChainSecurity_MakerDAO_zkSync_DAI_Bridge_audit.pdf

[208] “Using with upgrades - openzeppelin docs.” [Online]. Available: https://docs.openzeppelin.com/contracts/3.x/upgradeable

[209] 0xriptide, “Hackers in arbitrum’s inbox,” Sep 2022. [Online]. Available: https://medium.com/@0xriptide/hackers-in-arbitrums-inbox-ca23272641a2

[210] S. Singh Sidhu, M. N. H. Nguyen, C. Ngene, and S. Rouhani, “Trust development for blockchain interoperability using self-sovereign identity integration,” in *2022 IEEE 13th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, Oct 2022, p. 0033–0040.

[211] H. Kalodner, S. Goldfeder, X. Chen, S. M. Weinberg, and E. W. Felten, “Arbitrum: scalable, private smart contracts,” in *Proceedings of the 27th USENIX Conference on Security Symposium*, ser. SEC’18. USA: USENIX Association, Aug 2018, p. 1353–1370.

[212] A. Rondelet and Q. Kilbourn, “Threshold encrypted mempools: Limitations and considerations,” 2023.

[213] K. Qin, L. Zhou, and A. Gervais, “Quantifying Blockchain Extractable Value: How dark is the forest?” Dec. 2021, arXiv:2101.05511 [cs]. [Online]. Available: http://arxiv.org/abs/2101.05511

[214] Mar 2023. [Online]. Available: https://cointelegraph.com/news/arbitrum-discord-hacker-shares-phishing-announcement-amid-airdrop-hype

[215] H. Chen, M. Pendleton, L. Njilla, and S. Xu, “A Survey on Ethereum Systems Security: Vulnerabilities, Attacks, and Defenses,” *ACM Computing Surveys*, vol. 53, no. 3, pp. 67:1–67:43, Jun. 2020. [Online]. Available: https://doi.org/10.1145/3391195

[216] F. Barbàra and C. Schifanella, “Mp-htlc: Enabling blockchain interoperability through a multiparty implementation of the hash timelock contract,” *Concurrency and Computation: Practice and Experience*, vol. 35, no. 9, p. e7656, 2023.

[217] “Slither: the Solidity source analyzer,” May 2023. [Online]. Available: https://github.com/crytic/slither

[218] “Ronin bridge.” [Online]. Available: https://bridge.roninchain.com/

[219] “Polybridge.” [Online]. Available: https://bridge.poly.network/

[220] “Binance.” [Online]. Available: https://www.binance.org/

[221] “Nomad | bridge.” [Online]. Available: https://app.nomad.xyz/

[222] “Bxh.” [Online]. Available: https://app.bxh.com/#/

[223] “Multichain - cross chain router protocol.” [Online]. Available: https://app.multichain.org/#/router

[224] “Harmony one-eth bridge.” [Online]. Available: https://bridge.harmony.one/one

[225] “Qubit.” [Online]. Available: https://xbridge.qbt.fi

[226] “ptokens dapp.” [Online]. Available: https://dapp.ptokens.io/#/swap?asset=btc&from=btc&to=eth

[227] “Thorchain.” [Online]. Available: https://thorchain.org/

[228] “Meter passport.” [Online]. Available: https://passport.meter.io/#/

[229] “Chainswap.” [Online]. Available: https://exchange.chainswap.com/#/dashboard

[230] D. Tutku, “How fall down crypto hacks drop in 2023,” May 2023. [Online]. Available: https://medium.com/coinmonks/how-fall-down-crypto-hacks-drop-in-2023-a57b6c193f0d

[231] [Online]. Available: https://immunefi.com/explore/?filter=productType%3DCrosschain%2BLiquidity

[232] “List of btc addresses controlled by the pnetwork attacker.” [Online]. Available: https://pastebin.com/raw/bAquZVws

[233] “Polynetwork and hacker communicate.” [Online]. Available: https://docs.google.com/spreadsheets/u/1/d/11LUJwLoHX8ZCyfjhg5YZ0V99iU6PafMNL_NET45FSVc

[234] C. Team, “Poly network attacker returning funds after pulling off biggest defi theft ever,” Aug 2021. [Online]. Available: https://blog.chainalysis.com/reports/poly-network-hack-august-2021/

[235] “Circle rolls out native usdc tokens on polygon,” Oct. 2023. [Online]. Available: https://cointelegraph.com/news/circle-launches-usdc-tokens-on-polygon

[236] B. Putz, F. Böhm, and G. Pernul, *HyperSec: Visual Analytics for Blockchain Security Monitoring*, ser. IFIP Advances in Information and Communication Technology. Cham: Springer International Publishing, 2021, vol. 625, p. 165–180. [Online]. Available: https://link.springer.com/10.1007/978-3-030-78120-0_11

[237] S. M. Werner, D. Perez, L. Gudgeon, A. Klages-Mundt, D. Harz, and W. J. Knottenbelt, “SoK: Decentralized Finance (DeFi),” Sep. 2022, arXiv:2101.08778 [cs, econ, q-fin]. [Online]. Available: http://arxiv.org/abs/2101.08778

[238] T. Krupa, M. Ries, I. Kotuliak, K. Košťál, and R. Bencel, “Security Issues of Smart Contracts in Ethereum Platforms,” in *2021 28th Conference of Open Innovations Association (FRUCT)*, Jan. 2021, pp. 208–214, iSSN: 2305-7254.

[239] P. M. Caversaccio, "A historical collection of reentrancy attacks," May 2023, accessed on 12.09.2023. [Online]. Available: https://github.com/pcaversaccio/reentrancy-attacks

[240] "Mythril: Security analysis tool for EVM bytecode." [Online]. Available: https://github.com/ConsenSys/mythril

[241] "Mythx: Smart contract security service for ethereum." [Online]. Available: https://mythx.io/

[242] P. Tolmach, Y. Li, S.-W. Lin, Y. Liu, and Z. Li, "A survey of smart contract formal specification and verification," *ACM Comput. Surv.*, vol. 54, no. 7, jul 2021. [Online]. Available: https://doi.org/10.1145/3464421

[243] "Echidna: A fast smart contract fuzzer," May 2023. [Online]. Available: https://github.com/crytic/echidna

[244] V. Wüstholz and M. Christakis, "Harvey: a greybox fuzzer for smart contracts," in *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. Virtual Event USA: ACM, Nov 2020, p. 1398–1409. [Online]. Available: https://dl.acm.org/doi/10.1145/3368089.3417064

[245] "Scribble," May 2023. [Online]. Available: https://github.com/ConsenSys/scribble

[246] "Openzeppelin/openzeppelin-contracts," May 2023. [Online]. Available: https://github.com/OpenZeppelin/openzeppelin-contracts

[247] "A review and analysis of bridge hacks 2022," Oct 2022. [Online]. Available: https://crosschainbridges.blog/2022/10/13/analyzing-bridge-hacks/

[248] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. Los Alamitos, CA, USA: IEEE Computer Society, oct 2001, p. 136. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/SFCS.2001.959888

[249] L. Lamport, "Specifying systems: the tla+ language and tools for hardware and software engineers," 2002.

[250] Immunefi, "Polygon double-spend bug fix postmortem — $2m bounty," Feb 2023. [Online]. Available: https://medium.com/immunefi/polygon-double-spend-bug-fix-postmortem-2m-bounty-5a1db09db7f1

[251] P. Ladisa, H. Plate, M. Martinez, and O. Barais, "Sok: Taxonomy of attacks on open-source software supply chains," in *2023 2023 IEEE Symposium on Security and Privacy (SP) (SP)*. Los Alamitos, CA, USA: IEEE Computer Society, may 2023, pp. 167–184. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/SP46215.2023.00010

[252] "Celer network." [Online]. Available: https://celer.network/

[253] E. E. G. Peter Robinson. (2023) Erc 20 bridge security. Accessed on 16 October 2023. [Online]. Available: https://www.youtube.com/watch?v=hGDH6CNuMM0&t=580s

[254] G. Verdian, P. Tasca, C. Paterson, and G. Mondelli, "Quant overledger whitepaper," *Release V0*, vol. 1, p. 31, 2018.

[255] C. McMenamin, "Sok: Cross-domain mev," 2023.

[256] R. Belchior, "Dlt interoperability and more 28 — sok: Cross-domain mev," Sep. 2023. [Online]. Available: https://pt.linkedin.com/posts/rafaelpbelchior_blockchain-interoperability-blockdaemon-activity-7058963415154778112-47Ay

[257] E. C. Bank. (2023) Eurosystem proceeds to next phase of digital euro project. Accessed on 16 October 2023. [Online]. Available: https://www.ecb.europa.eu/press/pr/date/2023/html/ecb.pr231018~111a014ae7.en.html

[258] R. Yin, Z. Yan, X. Liang, H. Xie, and Z. Wan, "A survey on privacy preservation techniques for blockchain interoperability," *Journal of Systems Architecture*, p. 102892, Apr 2023.

[259] B. Pillai, K. Biswas, Z. Hóu, and V. Muthukkumarasamy, "Level of conceptual interoperability model for blockchain based systems," in *2022 IEEE Crosschain Workshop (ICBC-CROSS)*. IEEE, 2022, pp. 1–7.

[260] R. Ganguly, Y. Xue, A. Jonckheere, P. Ljung, B. Schornstein, B. Bonakdarpour, and M. Herlihy, "Distributed runtime verification of metric temporal properties for cross-chain protocols," in *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*, 2022, pp. 23–33.

[261] C. Giulio, "Before ethereum. the origin and evolution of blockchain oracles." *IEEE Access*, pp. 1–1, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10131932/

[262] K. E. Moujahid. (2022, November) Introducing a low-latency oracle solution for the defi derivatives market. Accessed on 16 October 2023. [Online]. Available: https://blog.chain.link/low-latency-oracle-solution/

[263] 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 Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1977–2008, 2020.

[264] J. Garay, A. Kiayias, and N. Leonardos, "The bitcoin backbone protocol: Analysis and applications," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2015, pp. 281–310.

[265] ——, "The bitcoin backbone protocol with chains of variable difficulty," in *Annual International Cryptology Conference*. Springer, 2017, pp. 291–323.

[266] R. Pass, L. Seeman, and A. Shelat, "Analysis of the blockchain protocol in asynchronous networks," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2017, pp. 643–673.

[267] W. Y. M. M. Thin, N. Dong, G. Bai, and J. S. Dong, "Formal analysis of a proof-of-stake blockchain," in *2018 23rd International Conference on Engineering of Complex Computer Systems (ICECCS)*. IEEE, 2018, pp. 197–200.

[268] M. Graf, R. Küsters, and D. Rausch, "Accountability in a permissioned blockchain: Formal analysis of hyperledger fabric," in *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2020, pp. 236–255.

[269] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *Annual international cryptology conference*. Springer, 2017, pp. 357–388.

[270] R. Zhang, R. Xue, and L. Liu, "Security and privacy on blockchain," *ACM Computing Surveys (CSUR)*, vol. 52, no. 3, pp. 1–34, 2019.

[271] J. Liu and Z. Liu, "A survey on security verification of blockchain smart contracts," *IEEE Access*, vol. 7, pp. 77 894–77 904, 2019.

[272] J. F. Ferreira, P. Cruz, T. Durieux, and R. Abreu, "Smartbugs: A framework to analyze solidity smart contracts," in *Proceedings of the 35th IEEE/ACM international conference on automated software engineering*, 2020, pp. 1349–1352.

[273] P. J. Taylor, T. Dargahi, A. Dehghantanha, R. M. Parizi, and K.-K. R. Choo, "A systematic literature review of blockchain cyber security," *Digital Communications and Networks*, vol. 6, no. 2, pp. 147–156, 2020.

[274] B. Putz and G. Pernul, "Detecting Blockchain Security Threats," in *2020 IEEE International Conference on Blockchain (Blockchain)*, Nov. 2020, pp. 313–320.

[275] J. Leng, M. Zhou, J. L. Zhao, Y. Huang, and Y. Bian, "Blockchain security: A survey of techniques and research directions," *IEEE Transactions on Services Computing*, vol. 15, no. 4, pp. 2490–2510, 2020.

[276] L. Zhou, X. Xiong, J. Ernstberger, S. Chaliasos, Z. Wang, Y. Wang, K. Qin, R. Wattenhofer, D. Song, and A. Gervais, "Sok: Decentralized finance (defi) attacks," 2023.

[277] N. Atzei, M. Bartoletti, and T. Cimoli, "A survey of attacks on ethereum smart contracts (sok)," in *Principles of Security and Trust: 6th International Conference, POST 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings 6*. Springer, 2017, pp. 164–186.

[278] W. Li, J. Bu, X. Li, and X. Chen, "Security analysis of defi: Vulnerabilities, attacks and advances," in *2022 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 2022, pp. 488–493.

[279] A. Pfitzmann and M. Hansen, "A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management," Dresden, Germany, Tech. Rep., 2010, accessed on 16 October 2023. [Online]. Available: http://www.maroki.de/pub/dphistory/2010_Anon_Terminology_v0.34.pdf

[280] G. Kelly, B. McKenzie *et al.*, "Security, privacy, and confidentiality issues on the internet," *Journal of Medical Internet Research*, vol. 4, no. 2, p. e861, 2002.

[281] J. B. Bernabe, J. L. Canovas, J. L. Hernandez-Ramos, R. T. Moreno, and A. Skarmeta, "Privacy-preserving solutions for blockchain: Review and challenges," *IEEE Access*, vol. 7, pp. 164 908–164 940, 2019.

[282] E. Androulaki, A. Barger, V. Bortnikov, S. Muralidharan, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Murthy, C. Ferris, G. Laventman, Y. Manevich, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, and J. Yellick, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proceedings of the 13th EuroSys conference, EuroSys 2018*, vol. 2018-Janua. New York, New York, USA: Association for Computing Machinery, Inc, Apr. 2018, pp. 1–15.

[283] N. R. Haddaway, M. J. Page, C. C. Pritchard, and L. A. McGuinness, "Prisma2020: An r package and shiny app for producing prisma 2020-compliant flow diagrams, with interactivity for optimised digital transparency and open synthesis," *Campbell Systematic Reviews*, vol. 18, no. 2, p. e1230, Jun 2022.

[284] R. Zhang, R. Xue, and L. Liu, "Security and Privacy on Blockchain," *ACM Computing Surveys*, vol. 52, no. 3, pp. 1–34, May 2020. [Online]. Available: https://dl.acm.org/doi/10.1145/3316481

[285] R. Belchior, "Privacy-preserving htlc in hyperledger cacti," 2023, accessed: 16-October-2023. [Online]. Available: https://github.com/hyperledger/cacti/blob/main/packages/cactus-plugin-htlc-eth-besu/src/main/solidity/contracts/PrivateHashTimeLock.sol

[286] E. Abebe, Y. Hu, A. Irvin, D. Karunamoorthy, V. Pandit, V. Ramakrishna, and J. Yu, "Verifiable observation of permissioned ledgers," in *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, May 2021, p. 1–9.

[287] J. R. Douceur, "The sybil attack," in *Peer-to-Peer Systems*, ser. Lecture Notes in Computer Science, P. Druschel, F. Kaashoek, and A. Rowstron, Eds. Berlin, Heidelberg: Springer, 2002, p. 251–260.

[288] Y. Liu, Q. Lu, L. Zhu, H.-Y. Paik, and M. Staples, "A systematic literature review on blockchain governance," *Journal of Systems and Software*, vol. 197, p. 111576, 2023.

[289] X. Fan, Q. Chai, and Z. Zhong, "Multav: A multi-chain token backed voting framework for decentralized blockchain governance," in *International Conference on Blockchain*. Springer, 2020, pp. 33–47.

[290] L. Huo, A. Klages-Mundt, A. Minca, F. C. Münter, and M. R. Wind, "Decentralized governance of stablecoins with closed form valuation," in *The International Conference on Mathematical Research for Blockchain Economy*. Springer, 2022, pp. 59–73.

[291] "Allbridge is the best cross-chain bridging solution provider." [Online]. Available: https://keen-newton-441bbb.netlify.app/

[292] A. Obadia, A. Salles, L. Sankar, T. Chitra, V. Chellani, and P. Daian, "Unity is strength: A formalization of cross-domain maximal extractable value," *arXiv preprint arXiv:2112.01472*, 2021.

[293] A. Judmayer, N. Stifter, P. Schindler, and E. Weippl, "Estimating (miner) extractable value is hard, let's go shopping!" *Cryptology ePrint Archive*, 2021.

[294] F. Kamphuis, B. Magri, R. Lamberty, and S. Faust, "Revisiting transaction ledger robustness in the miner extractable value era," in *International Conference on Applied Cryptography and Network Security*. Springer, 2023, pp. 675–698.

[295] A. Klages-Mundt, D. Harz, L. Gudgeon, J.-Y. Liu, and A. Minca, "Stablecoins 2.0: Economic Foundations and Risk-based Models," in *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, ser. AFT '20. New York, NY, USA: Association for Computing Machinery, Oct. 2020, pp. 59–79. [Online]. Available: https://doi.org/10.1145/3419614.3423261

[296] A. Judmayer, N. Stifter, A. Zamyatin, I. Tsabary, I. Eyal, P. Gaži, S. Meiklejohn, and E. Weippl, "Pay to win: Cheap, cross-chain bribing attacks on pow cryptocurrencies," in *Financial Cryptography and Data Security. FC 2021 International Workshops: CoDecFin, DeFi, VOTING, and WTSC, Virtual Event, March 5, 2021, Revised Selected Papers 25*. Springer, 2021, pp. 533–549.

[297] T. Nadahalli, M. Khabbazian, and R. Wattenhofer, "Timelocked bribing," in *Financial Cryptography and Data Security: 25th International Conference, FC 2021, Virtual Event, March 1–5, 2021, Revised Selected Papers, Part I 25*. Springer, 2021, pp. 53–72.

[298] K. Kulkarni, T. Diamandis, and T. Chitra, "Towards a theory of maximal extractable value i: Constant function market makers," *arXiv preprint arXiv:2207.11835*, 2022.

[299] Immunefi, "Wormhole uninitialized proxy bugfix review," Feb 2023. [Online]. Available: https://medium.com/immunefi/wormhole-uninitialized-proxy-bugfix-review-90250c41a43a

[300] T. Pornin, "Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)," RFC 6979, Aug. 2013. [Online]. Available: https://www.rfc-editor.org/info/rfc6979

[301] J. Groth, "On the size of pairing-based non-interactive arguments," in *Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II 35*. Springer, 2016, pp. 305–326.

[302] (2023) Supported chain ids - endpoint contracts. Accessed on 16 October 2023. [Online]. Available: https://layerzero.gitbook.io/docs/technical-reference/mainnet/supported-chain-ids

[303] "Rekt - ronin network." [Online]. Available: https://rekt.news/ronin-rekt/

[304] R. Behnke, "Explained: The ronin hack (march 2022)," Mar 2022. [Online]. Available: https://www.halborn.com/blog/post/explained-the-ronin-hack-march-2022

[305] "Rekt - poly network." [Online]. Available: https://rekt.news/polynetwork-rekt/

[306] B. A. [@BeosinAlert], "Polynetwork2 have suffered a potential compromise of private keys or a multi-signature service attack. the hacker has exploited forged proofs to initiate withdrawal operations on the cross-chain bridge contracts across multiple chains. an analysis thread," Jul 2023. [Online]. Available: https://twitter.com/BeosinAlert/status/1675708122944483328

[307] "Rekt - poly network - rekt 2." [Online]. Available: https://rekt.news/polynetwork-rekt2/

[308] D. [@dedaub], "Getting to the bottom of the "34 billion" poly network hack with a technical postmortem. tl; dr poly network had a simple 3 of 4 multisig arrangement over 2 years! looking at the final event we found that the private keys to the addresses marked were compromised. https://t.co/y0emjxcyso," Jul 2023. [Online]. Available: https://twitter.com/dedaub/status/1675516729349292032

[309] [Online]. Available: https://www.rekt.news/poly-network-rekt2/

[310] samczsun [@samczsun], "Five hours ago, an attacker stole 2 million bnb ($566m usd) from the binance bridge. during that time, i've been working closely with multiple parties to triage and resolve this issue. here's how it all went down. https://t.co/e0885dc3lw," Oct 2022. [Online]. Available: https://twitter.com/samczsun/status/1578167198203289600

[311] "Rekt - wormhole." [Online]. Available: https://rekt.news/wormhole-rekt/

[312] "Wormhole bridge exploit incident analysis - blog - web3 security leaderboard." [Online]. Available: https://certik.com/resources/blog/1kDYgyBcisoD2EqiBpHE5l-wormhole-bridge-exploit-incident-analysis

[313] "Nomad bridge incident analysis." [Online]. Available: https://www.coinbase.com/blog/nomad-bridge-incident-analysis

[314] Q. [@Quantstamp], "The exact bug that led to the exploit was in commit 46d145, which introduced new logic that was not part of the audit. https://t.co/k00my1sg1u," Aug 2022. [Online]. Available: https://twitter.com/Quantstamp/status/1554348522656256001

[315] "Harmony incident analysis - blog - web3 security leaderboard." [Online]. Available: https://certik.com/resources/blog/2QRuMEEZAWHx0f16kz43uC-harmony-incident-analysis

[316] "Rekt - harmony bridge." [Online]. Available: https://www.rekt.news/harmony-rekt/

[317] Elliptic, "The harmony horizon bridge hack." [Online]. Available: https://www.elliptic.co/hubfs/Harmony%20Horizon%20Bridge%20Hack%20P1%20briefing%20note%20final.pdf

[318] "Rekt - qubit finance." [Online]. Available: https://rekt.news/qubit-rekt/

[319] [Online]. Available: https://thearchitect.notion.site/THORChain-Incident-07-15-7d205f91924e44a5b6499b6df5f6c210

[320] "Rekt - thorchain - rekt 2." [Online]. Available: https://rekt.news/thorchain-rekt/

[321] Lossless, "Thorchain hacks — could they have been prevented?" Aug 2021. [Online]. Available: https://losslessdefi.medium.com/thorchain-hacks-could-they-have-been-prevented-6e4e478d0831

[322] "Rekt - thorchain - rekt 2." [Online]. Available: https://rekt.news/thorchain-rekt2/

[323] R. Behnke, "Explained: The thorchain hack (july 2021)," Jul 2021. [Online]. Available: https://www.halborn.com/blog/post/explained-the-thorchain-hack-july-2021

[324] ChainSwap, "Chainswap exploit 11 july 2021 post-mortem," Jul 2021. [Online]. Available: https://chain-swap.medium.com/chainswap-exploit-11-july-2021-post-mortem-6e4e346e5a32

[325] "Rekt - chainswap." [Online]. Available: https://www.rekt.news/

[326] R. Behnke, "Explained: The pnetwork hack (september 2021)," Oct 2021. [Online]. Available: https://www.halborn.com/blog/post/explained-the-pnetwork-hack-september-2021

[327] p. Team, "pnetwork post mortem: pbtc-on-bsc exploit," Sep 2021. [Online]. Available: https://medium.com/pnetwork/pnetwork-post-mortem-pbtc-on-bsc-exploit-170890c58d5f

[328] M. P. Anyswap), "Anyswap multichain router v3 exploit statement," Jul 2021. [Online]. Available: https://medium.com/multichainorg/anyswap-multichain-router-v3-exploit-statement-6833f1b7e6fb

[329] nick.eth [@nicksdjohnson], "In case you were wondering if anyswap is safe now they've patched the bug, i present for your consideration, the patch: https://t.co/c3fiawxi4l." [Online]. Available: https://twitter.com/nicksdjohnson/status/1414512086672052238

[330] M. [@MultichainOrg], "1. on may 21, 2023, multichain ceo zhaojun was taken away by the chinese police from his home and has…," Jul 2023. [Online]. Available: https://twitter.com/MultichainOrg/status/1679768407628185600

[331] E. Gkritsi, "$139m bxh exchange hack was the result of leaked admin key," Nov 2021. [Online]. Available: https://www.coindesk.com/tech/2021/11/01/139m-bxh-exchange-hack-was-the-result-of-leaked-admin-key/

[332] R. Behnke, "Explained: The bxh exchange hack (october 2021)," Nov 2021. [Online]. Available: https://www.halborn.com/blog/post/explained-the-bxh-exchange-hack-october-2021

Appendix

1. Cross-Chain Concept Formalization

We represent a local transaction in one blockchain as $t = \langle id, ts, target, payload, \sigma_{k_i}(id, ts, target, payload)\rangle$, where *id* is the local transaction identifier, *ts* is the transaction timestamp, *target* is the state key to which the transaction refers, *payload* is the transaction payload, and $\sigma_{k_i}(id, ts, target, payload)$ is the signature issued party i, the initiator of the transaction. A transaction t is considered final in a ledger l according to a security parameter λ of that network (e.g., the block containing the transaction has a minimum height), and is represented as $final^{l}(t) \rightarrow \{0, 1\}$.

A local transaction yields a state change in the form of a key-value pair. We represent a state change as $s(t) = \langle s_k, s_{k,v}\rangle$, where s_k corresponds to the *target* of t, and $s_{k,v}$ its new value. The execution of local transactions emits events. Events act as labels or wrappers for state changes caused by local transactions. As an example, a local event targeting transaction t is represented as $\langle t_{id}, type, store\rangle$, where t_{id} represents the identifier of the transaction t, $type$ is the type of state change (e.g., the lock of an asset), and $store$ is a key-value store representing the new state after executing t.

Definition 10 (Cross-Chain Event)**.** *A ccevent gives a cross-chain meaning to a local event. It extends a local event with metadata, representing a state change in a certain ledger. We denote* $e_{type}^{l \in \mathcal{L}}(t)$ *a cross-chain event that represents a state change of type type against* $t.target$*, in domain* l*, emitted by transaction* t*, such that* $final^{l}(t) = 1$.

In our model, a *ccevent* is only created when the transaction that emits the corresponding local event is considered final. However, it might be valid or not according to $\mathcal{R}$ that defines the expected behaviour.

Definition 11 (Valid Cross-Chain Event)**.** *A cross-chain event* $e_{type}^{l_1 \in \mathcal{L}}(t)$ *is deemed valid if and only if it follows the defined cross-chain rules related to it.*

Note that the validity of an event emitted by a local transaction does not imply the validity of the corresponding cross-chain event because the latter might not comply with the defined cross-chain rules.

Formally, a *cctx* is then a composition of n ordered cross-chain events $\mathcal{E}$ across multiple ledgers $\mathcal{L}$ with the same $cctxid$, such that $\mathcal{E} = \{e_{type_1}^{l_1 \in \mathcal{L}}(t_1), e_{type_2}^{l_2 \in \mathcal{L}}(t_2), ..., e_{type_n}^{l_k \in \mathcal{L}}(t_i)\}$. The validity of a cross-chain transaction is given by the conjunction of the validity of every cross-chain event in $\mathcal{E}$, that is evaluated against a set of rules $\mathcal{R}$. We consider blockchain rules $\mathcal{R}$ to be a composition of predicates $\zeta = \{\zeta_1(\mathcal{E}), \zeta_2(\mathcal{E}), ..., \zeta_n(\mathcal{E})\}$ over a set of events $\mathcal{E}$.

Cross-chain models are a set of metrics, that evaluate $cctxs$ against cross-chain rules, formalized in [32].

2. Survey Methodology

In this section, we present further details on our systematic survey methodology. Figure 8 presents the PRISMA diagram for our survey.

2.1. Data Sources. Blockchain interoperability research has been rapidly evolving in the last couple of years. Yet, academic and peer-reviewed work alone falls short of delivering the most up-to-date facts on interoperability, particularly in the analysis of cross-chain hacks. We pay attention to a significant amount of material available as grey literature in online databases such as *Rekt* and *Slowmist*, and online audit reports by reputed companies in the area such as *Certik*, *Chainsecurity* and *Trail of Bits*. We also find that many incident reports are divulged through unstructured and informal means of communication, namely blog or social media posts [28]. We strive to uphold the integrity of the findings presented in this work, diligently cross-referencing information whenever possible. Therefore, to the best of our knowledge, the material presented is the most reliable and up-to-date.

2.2. Threat Model Taxonomy. In this study, we present vulnerabilities identified in multiple contexts. Sometimes

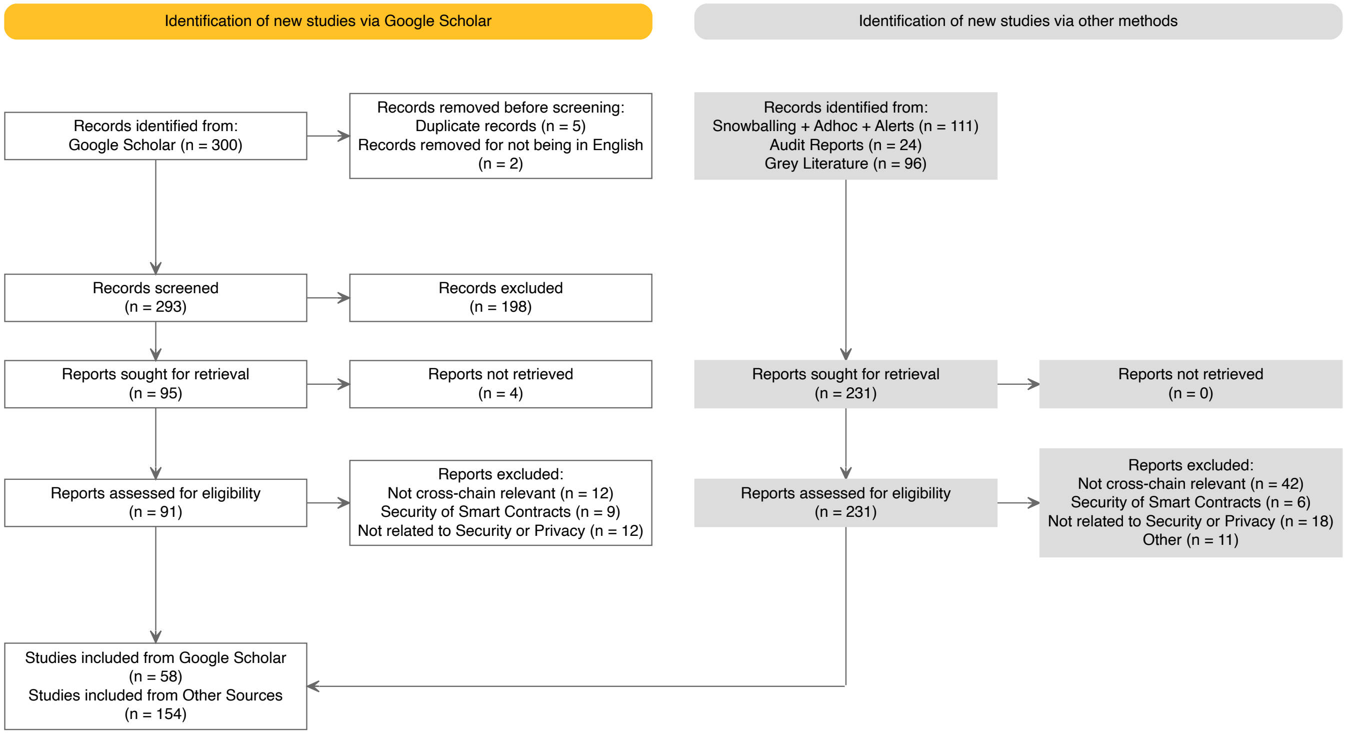

Identification of new studies via Google Scholar
Identification of new studies via other methods
Records identified from: Google Scholar (n = 300)
Records removed before screening: Duplicate records (n = 5) Records removed for not being in English (n = 2)
Records identified from: Snowballing + Adhoc + Alerts (n = 111) Audit Reports (n = 24) Grey Literature (n = 96)
Records screened (n = 293)
Records excluded (n = 198)
Reports sought for retrieval (n = 95)
Reports not retrieved (n = 4)
Reports sought for retrieval (n = 231)
Reports not retrieved (n = 0)
Reports assessed for eligibility (n = 91)
Reports excluded: Not cross-chain relevant (n = 12) Security of Smart Contracts (n = 9) Not related to Security or Privacy (n = 12)
Reports assessed for eligibility (n = 231)
Reports excluded: Not cross-chain relevant (n = 42) Security of Smart Contracts (n = 6) Not related to Security or Privacy (n = 18) Other (n = 11)
Studies included from Google Scholar (n = 58) Studies included from Other Sources (n = 154)

Figure 8. PRISMA Diagram depicting our methodology [283]

these vulnerabilities were seen in previous attacks, audit reports, bug bounty reports, or academic papers. Attacks encompass every action, intentional or not, that breaks the liveness or safety of a cross-chain protocol. An attacker is an entity that performs an attack and might be motivated either to increase profit or just to harm the system, having an arbitrary behavior.

2.3. Paper Classification. Out of the chosen papers, only a fraction were categorized according to the specified security and privacy attributes. Specifically, 51 papers are classifiable due to their inclusion of security-enhancing or privacy-preserving solutions. However, the remaining papers cannot be classified since most are either surveys or concentrate on modeling architectures.

3. Crpytographic tools

We refer the reader to the formalization of cryptographic primitives such as digital signatures, blind signatures, Adaptor Signatures and non-interactive proof systems (Appendix A of [105]). We highlight the importance of other building blocks to construct secure interoperability solutions. Namely, we assume protocols use pre-image resistant hash functions, unforgeable digital signature schemes, and trusted communication channels (e.g., TLS) between blockchain nodes and bridge operators.

3.1. Multi Party Computation. Multi-party computation (MPC) is a widely used cryptographic primitive that allows multiple untrusted parties to perform joint computation on participants' private inputs [103]. Consider a group of participants $p_1, ..., p_n$, where every participant p_i has a piece of data d_i that must remain private. MPC allows all participants to publicly compute the value of a function $f(d_1, ..., d_n)$, while d_i remains private to each party. A higher level of security is achieved by combining MPC with other cryptographic primitives, such as secret-sharing schemes – e.g., Shamir Secret Sharing (CITE). These allow a secret key S to be split into multiple fragments (also called shares) $s_1, ..., s_n$ that are randomly distributed between the participants. This way, no party has the ability to perform operations on its own because it does not own the whole key.

3.2. Threshold Signature Schemes. Using threshold signature schemes (TSS), a secret S can only be reconstructed if a threshold of the n parties collaborate. The secret S can be used to sign a message on behalf of a whole group, without each individual party revealing their secret shares. In a *(m,n)-threshold signature scheme*, assuming that there are at most $m-1$ dishonest parties, m parties are enough for a signature to be deemed valid.

4. Privacy Properties – Example

For simplicity let us consider a cross-chain transaction composed of two local transactions t_1 and t_2 in ledgers l_1 and l_2, respectively. The first transaction locks an asset X in l_1 by transferring it from the user $\mathcal{E}_1$ to the bridge contract $\mathcal{E}_2$, whereas the second transaction transfers (or mints) a representation of that asset from the bridge contract to the user's address in the target chain. Additionally, consider $t = (\mathcal{E}_1, \mathcal{E}_2, payload, l)$ a local transaction on ledger l. The above $cctx$ can then be represented by $\langle(\mathcal{E}_1, \mathcal{E}_2, payload, l_1), (\mathcal{E}_2, \mathcal{E}_1, payload, l_2)\rangle$. Cross-chain anonymity guarantees that for any parties $\mathcal{E}'_1$ and $\mathcal{E}'_2$, two $cctxs$ between different addresses are indistinguishable $\approx$, i.e., $\langle(\mathcal{E}_1, \mathcal{E}_2, payload, l_1), (\mathcal{E}_2, \mathcal{E}_1, payload, l_2)\rangle \approx \langle(\mathcal{E}'_1, \mathcal{E}'_2, payload, l_1), (\mathcal{E}'_2, \mathcal{E}'_1, payload, l_2)\rangle$. This formalization abstracts the payload being transferred, and the role of the entities in the transfer. It can be an asset, a token, or general data, and the entities involved in the $cctx$ can be both final users, smart contracts, or the bridging entity. Bear in mind that if a user holds full anonymity in one chain, yet the interoperability mechanism does not guarantee the unlinkability of transactions, that degree of anonymity (cross-chain wise) decreases.

5. Confidentiality as a requirement for Cross-Chain Unlinkability

From our research, we show that the degree of linkability, and consequently anonymity, yielded by an interoperability solution is tied to the capacity to keep local transactions' content confidential. This idea is also supported by [284]. We study the requirements for cross-chain unlinkability to be achieved and propose Lemma A.1, which is proved below. We denote *confidential systems* as $\mathcal{C}$ and *non-confidential systems* as $\mathcal{S}$. Confidential systems are either private blockchains (e.g., Hyperledger Fabric, Quorum, DAML's Canton) or public blockchains that are hardened by a privacy-preserving mechanism that hides transactional data (e.g., ZCash, Monero). Non-confidential systems are typical layer-one blockchains with no concern over privacy (e.g., Ethereum, NEAR). We denote an interoperation process (e.g., asset transfer/data transfer) by →.

Lemma A.1. *Cross-chain unlinkability is unlikely to be achieved without confidentiality on the underlying chains.*

Proof. Interoperability inherently relies on linkability between transactions on a source and target blockchain. However, one must consider that this linkability is undesirable for general users, as it compromises the degree of privacy yielded by the protocol. We identify a direct relation between the confidentiality guarantees offered by one blockchain and the cross-chain anonymity and unlinkability offered by interoperability solutions built on top of these. Table E summarizes the comparison based on the privacy guarantees of the privacy guarantees of the ledgers, which is further explained below.

$\mathcal{S} \rightarrow \mathcal{C}$: Firstly, let us assume the source chain does not guarantee confidentiality, but the target chain does. On the source chain, transaction data (such as the amount, sender, and recipient) is open to everyone. However, since the destination chain is private or has privacy-preserving primitives, only a set of authorized parties can see and link this transaction to the one issued on the source chain (might be a whole blockchain, one party [154], or a set of parties that share a private channel [282]. To issue transactions on a permissioned network, there must be a trusted and identified IM with access to the ledger and, optionally, to private channels. Assuming trust in this entity it does not disclose this information, and therefore, external parties cannot link transactions. Transaction unlinkability is guaranteed.

TABLE 7. Maximum Achievable Cross-Chain Anonymity Based on the Cross-Chain unlinkability, which is dependent on the confidentiality of the underlying chains

Source Chain	Target Chain	Max. Unlinkability Achievable	Max. Anonymity Achievable
$\mathcal{S}$	$\mathcal{S}$	CC Linkability	CC Pseudonymity
$\mathcal{C}$	$\mathcal{S}$	CC Unlinkability	CC Anonymity
$\mathcal{S}$	$\mathcal{C}$	CC Unlinkability	CC Anonymity
$\mathcal{C}$	$\mathcal{C}$	CC Unlinkability	CC Anonymity

$\mathcal{C} \rightarrow \mathcal{S}$: Secondly, consider a private source chain and a non-private-concerned target blockchain. Transaction amounts and addresses are unknown to unauthorized parties within the source chain. The key challenge here is that there must be a way for the target blockchain to know if one action took place in the source one. The two options are a light client in the target chain, where the user presents a way of decrypting source blockchain data or an interoperability mechanism that has access to the blockchains and acts as a trusted party. In the former alternative, an option might be the usage of zero-knowledge proofs, which can be verified while maintaining data confidentiality. In the latter case, linkability is possible only if the trusted IM discloses information. A trusted IM can access this information, verify its validity and issue transactions on the public chain accordingly. Since the IM must be trusted, there is cross-chain unlinkability under these conditions.

$\mathcal{C} \rightarrow \mathcal{C}$: Assuming both blockchains are confidential, only authorized parties can link transactions, including a trusted IM with access credentials on both the source and target chain. Applying the same logic above, an external observer cannot link transactions in each chain.

$\mathcal{S} \rightarrow \mathcal{S}$: Assuming there is no confidentiality on the underlying ledgers – i.e., these are permissionless networks where information is widely open. Analysis of various heuristics, such as transaction amounts, addresses, or shared secrets, can enable linking transactions across multiple blockchains [82], [92]. Mixing services (cf. Section 5.2.1) help cover traces and break transaction linkability. In an ideal setting, these systems achieve their goals perfectly. However, in the real world, these have been studied and are shown not to be effective [145]. □

Note that there needs to be always some trusted party to enable interoperability. In centralized settings, this entity can hold records of transactions and corresponding mappings. Therefore, privacy concerns may arise, such as the leakage of private information or, in the worst-case scenario, sold [150]. With a trusted centralized entity that removes outdated records and does not keep track of transactions, there is full unlinkability without the risk of being compromised. A possible safeguard is cryptographic methods such as blind signatures, where a message is blindly signed by the trusted party (cf. Section). We derive a direct consequence from the above Lemma A.1: since cross-chain anonymity depends on the unlinkability of *cctxs*, we extend our initial thoughts in Lemma A.2.

Lemma A.2. *Cross-chain anonymity is unlikely to be achieved without confidentiality on the underlying chains.*

Proof. Cross-chain anonymity is driven by cross-chain unlinkability, and cross-chain unlinkability is unlikely to be achieved without the confidentiality of the underlying chains. Therefore, cross-chain anonymity is unlikely to be achieved without confidentiality on the underlying chains due to the incapacity to provide cross-chain unlinkability under those conditions. ☐

We derive the main consequence of the above ideas in Corollary A.2.1.

Corollary A.2.1. *The privacy level offered by the interoperability solution is upper-bounded by the intersection of the privacy levels of the underlying chains.*

6. Security Approaches – Extended Version

6.1. Permissionless Networks. We provide further explanations on atomic exchange protocols based on intermediary networks and on the design of the Blockchain Engines approach.

Decentralized atomic swap protocols, namely HTLCs and variants, require synchronous communication between parties. Therefore, they do not guarantee atomicity under longstanding crashes due to relying on a pre-defined timelock. Herlihy et al. [36] show that any atomic swap protocol that tolerates periods of asynchrony (i.e., under a semi-synchronous model) must rely on a third-party system or network that enforces the ordering of events. Hence, the authors propose the addition of a coordinator blockchain that counts *commit* and *abort* votes from the involved parties, who then extract proofs based on whether all parties voted to commit or abort the deal. Similarly, AC^{3}WN [80] also proposes a third-party witness blockchain that attests to the state of an atomic swap and decides whether it must be committed or aborted. The witness blockchain contains light client implementations for the supported chains and verifies inclusion proofs on the relayed block headers. The authors assume that participants relay block headers and are always valid if they follow the source chain consensus rules, which might not happen in fork-prone chains (cf. Section 4.1).

Blockchain Engines (or Blockchains of Blockchains) rely on a relay chain that provides shared security and composability in an interconnected environment. These networks are called zones, parachains and subnets in Cosmos [61], Polkadot [62] and Avalanche [115], respectively. All these projects have a custom messaging mechanism that allows arbitrary communication between networks within the same ecosystem, standardizing cross-chain communication within each ecosystem (we still need to address inter-ecosystem interoperability). These messaging protocols have different tradeoffs regarding customization and shared security with the main chain. Cosmos uses Inter Blockchain Communication (IBC), whereas Polkadot relies on Cross-Chain Message Passing (XCMP). IBC allows finer-grained control over the customization of the different zones. Therefore, their security is dependent on the specific design and implementation. Because it was built on top of Tendermint, it only connects chains. Polkadot and XCMP do not have those constraints. Individual chains share state with the entire system, which acts as a shared security layer, and *cctxs* can go directly to the destination chain, which aims at providing more scalability. Avalanche's Warp Messaging mechanism relies on a shared state of the primary chain to maintain whitelisted validator sets of each custom subnet. Topos [60] achieves a higher level of security through the proposed zkVM, where validity is delegated to the proving system, allowing the removal of trust assumptions on the validators of the blockchains for state validity. In this case, all The Topos zkVM verifies the subnets' state transitions. Computational proofs are then publicly verifiable by users in other subnets, allowing seamless interoperability between networks in the ecosystem.

6.2. Inclusion Proofs. The first mechanism proposed to validate transactions of l_1 in l_2 relied on Merkle proofs. l_2 requires a light client implementation of l_1, which means that there is a way of verifying l_1's consensus mechanism within l_2 [58]. It also relies on external entities, the relayers, who only relay block headers from l_1 to l_2, without producing any kind of proof. It is easier to verify the block headers of PoW-based chains because consensus verification only depends on the block headers assuming a trusted state initialization (i.e., either the genesis block was set manually, or any other final block) [90]. On the other hand, Chains based on PoS require the current validators' keys to be available in the $l2$. Keys can be stored in $l2$ and constantly updated, or they can be submitted with every new block header [91]. Knowing the cost of storage on the destination chain, transaction fees, the size of the validator set (and consequently the size of all keys), and the periodicity of each validator set election, one can design a solution that best fits their requirements and needs. Westerkamp et al. [91] calculate the cost of storing information on-chain for the Ethereum sync committee (groups of 512 randomly elected validators every ~1 day which is used to validate Ethereum consensus) as being more than 300 USD[¶]. The authors also propose a cheaper solution which resubmits all public keys with each update, avoiding storage costs even though it increases the transaction size. Boneh–Lynn–Shacham (BLS) multi-signatures have been used in multiple protocols to generate an aggregated digital signature built to prove the consensus mechanism of the source chain on the target.

[¶] value updated on 27th June 2023 – ETH price: 1872.15 USD, Gas Price: 45 GWei

Both [51] and [91] propose light client protocols for interoperating proof of stake blockchains based on signature aggregation techniques, namely applying Boneh–Lynn–Shacham (BLS) multi-signatures.We reckon that the cost of these protocols is mainly driven by the construction of the aggregated signature of the sync committee. They have some potential liveness issues, due to requiring at least one block from each epoch (once every ~27 hours) to be submitted to the destination chain to guarantee the correct transition of validation sets. The authors do not calculate the probability of such an event.

6.3. Off-Chain Communication Channels.

Hashlocks & Timelocks. Hash Time-Locked Contract (HTLC) [10], [38], a commit-reveal scheme based on hash locks and time locks, is a decentralized protocol to exchange assets. Parties agree on certain parameters off-chain and have predefined periods in which they must act to complete the protocol. The involved participants also need to observe state changes directly, i.e., without an intermediary relaying information. The security of these protocols relies on both the cryptographic primitives employed (e.g., the cryptographic hash function used for the hash locks) and on the off-chain communication channel used.

HTLCs function as follows. Assume that entities $\mathcal{E}_1$ and $\mathcal{E}_2$ want to swap asset X for asset Y, which are in l_1 and l_2 respectively. Entity $\mathcal{E}_1$ starts by generating a secret s, and publishes a smart contract on l_1 with hashlock $h(s)$, where h is a collision-resistant hash function. This smart contract contains a withdrawal function that transfers X to $\mathcal{E}_2$ if the solution of the hashlock is provided – i.e. if a value v in which $h(v) = h(s)$. By the properties of collision-resistant hash functions, v must be the secret s. Additionally, there is a duration Δ_1 (the timelock) in which the asset can be claimed by $\mathcal{E}_2$, otherwise, asset X is returned to $\mathcal{E}_1$. Then, $\mathcal{E}_2$, verifies the deployment of the smart contract on l_1 and deploys a smart contract on l_2 with timelock Δ_2 and the reverse operation – transferring Y to $\mathcal{E}_1$ if s is provided. $\mathcal{E}_1$ starts by redeeming Y on l_2 within Δ_2, which reveals s to $\mathcal{E}_2$. $\mathcal{E}_2$ can now use the secret to redeem X from l_1. We refer the reader to [38] for details and discussions about the strengths and limitations of HTLCs [285].

The difference between the timelocks deployed by $\mathcal{E}_1$ and $\mathcal{E}_2$ (in this case $\Delta_1 - \Delta_2 = \Delta$) is key to guaranteeing the atomicity of the protocol. If Δ is too small, $\mathcal{E}_2$ might not have enough time to redeem X on the source chain. This timelock must account for the network communication time between parties, the finality times on both blockchains and possibly some network delay or downtime. However, if Δ is too high and $\mathcal{E}_2$ abandons the protocol just after $\mathcal{E}_1$ locked X on the source chain, X will remain locked for Δ_1. This is known as the *Sore Loser Attack* [95].

Some protocols have been proposed to solve the aforementioned issues. The majority alter the synchronous communication assumptions [80], [87], while the ones that retain that property focus on the usage of premiums [40], [95], [137]. A *premium* is a value staked as collateral before the execution of the actual protocol. It must be a value acceptable by the victim as a possible compensation for locking up assets for the duration of the protocol. Simultaneously, it needs to be small enough so that parties engage in the swap – i.e., accept the risk of losing this value. There are multiple game-theoretical analyses of HTLCs or simple variations such as [38], [40], [99], [138]. In particular, [138] proves that the protocol has a higher chance of being successful (instead of being aborted) under these collateralized models or employing dynamic exchange rate adjustments. However, since premiums are deployed before the actual protocol, they are also vulnerable to *Griefing Attacks* (even though these are usually much smaller amounts than the initial values to swap). XCC [98] proposes the usage of timelocks in an overcollateralized model, where the escrowed assets are only transferred to the vault (bridge contract on source chain) once the corresponding assets are transferred to the target chain. This has a clear benefit of not transferring ownership of assets directly to the escrow, completely relying on correct behaviour.

Time-Based Cryptography. Relying on explicit time intervals is challenging when each permissionless blockchain has different time management mechanisms, usually implemented at a very coarse grain level (in the order of hours or days). Therefore, primitives such as *Verifiable Timed Commitments* (VTC) [139] or *Verifiable Timed Signatures* (VTS) [140] can mitigate the problem above.

Considering parties $\mathcal{E}_1$ and $\mathcal{E}_2$, the VTC scheme allows a committer to compute a cryptographic commitment C of a value v and a difficulty level d, and prove to the verifier that it is possible to open C either by having v (revealed by the committer), or by executing 2^d sequential computation steps – i.e., if one party decides not to reveal the value behind the commitment, it can be brute forced by the victim in a predefined amount of time driven by d. The difference for vanilla HTLCs is that instead of having a hashlock and a timelock, there is a more powerful hashlock which can also be opened with a certain number of computation steps. The authors of [106] propose an HTLC-based protocol, where the value committed to is a hash pre-image just like in HTLCs, which extends VTC with ZK cryptography to prove arbitrary attributes for the timed commitment.

Signature-based Locks. On the other hand, VTS allow a committer to produce a commitment of a signature σ on a value, and prove to the verifier that the σ is valid and revealable in time T. [42] proposes an atomic swap protocol, extensible to multiple parties, where the sender and the recipient share ownership of smart contracts on the source and destination chain, through joint key generation algorithms. As such, parties can use TVS on previously generated signatures from jointly signed refund transactions, which allows one to abort a swap if no action is performed within T, using the brute force algorithm. This is done by learning the other party's key (or key share) and gaining full control over the escrow address. Additionally, the authors of [100] present a commit-reveal scheme for atomic swaps based on adaptor signatures. These are verifiable partial signatures, that allow the revealing of a secret once the full signature is published. We question the liveness guarantees of the protocol if one party halts participation midway. Li et al. [104], also leverage adaptor-signatures and the involved parties first share and pre-sign revoke transactions such that one party can successfully abandon the protocol if the other halts participation. Finally, the security of Sweep-UC [105] is given by the security of a blind signature protocol between users and a third party who issues them. We also question the liveness of the solution since there is no incentive mechanism for the third party to engage in the protocol. Nevertheless, we acknowledge that these solutions pave the way for interoperability in script-minimal blockchains, with only signature verification functionality.

7. Privacy Approaches – Extended Version

7.1. Blind Signature Protocol. We provide an example of a suitable protocol using blind signatures.

1) The client generates a message m and applies a blind factor value, that originates m'.
2) The client sends the blinded message m' to the IM, who signs it, producing a blind signature $s(m')$.
3) The IM issues a transaction to the source chain, locking 10 tokens, and sends the blinded signature $s(m')$ to the client.
4) The client can unblind the signature using the blind factor previously applied to obtaining a valid signature on the original message, $s(m)$.
5) This signature can now be presented as proof of locking 10 tokens so that the equivalent is minted in the target chain.
6) The IM must keep track of already accepted messages so that blind signatures are not used to trigger multiple transactions on the target chain. If it haa not been accepted previously, the IM triggers a transaction minting 10 tokens in the target chain.

7.2. Homomorphic Encryption-based Protocol. We present a concrete example. [102] considers a homomorphic hash function h that satisfies $h(s_1) + h(s_2) = h(s_1 + s_2)$, for any 256 bit values s_1 and s_2. Alice generates secrets $s1$ and $s2$, and sends to Bob $h(1)$, $h(2)$, and $S = s1 + s_2$. Alice deploys a smart contract with hashlock $h(s_1)$, which is easily verifiable by Bob, given that Bob was sent $h(s1)$. In turn, Bob deploys a smart contract in the other blockchain with hashlock $h(s_2)$. Therefore, Alice can redeem Bob's funds by presenting $s2$. Bob accepts this given that he can now compute $s_1 = S - s_2$, which hash needs to match $h(s_1)$ received initially from Alice, allowing him to redeem Alice's assets. By deploying both smart contracts with different values, they are no longer linkable unless S is disclosed.

7.3. Anonymity based on Group Signatures. Much like Blind Signatures, Group Signatures (GS) also rely on a centralized party. In this scheme, users can sign messages on behalf of a group, with the centralized authority assuming the role of a group manager [144]. When utilizing this system, anyone possessing the group's public key can efficiently verify and authenticate signatures confirming that they originate from a group member. However, it does not reveal who signed the message. Importantly, the group manager wields the authority to oversee group membership, enabling actions like revoking membership or disclosing a signer in case of misconduct or, for instance, for auditing purposes. This capability to revoke anonymity becomes crucial when anonymity is worked for illicit purposes. GS safeguard anonymity while ensuring accountability, which holds significance for various entities, including government bodies, bolstering their trust in the technology. This capacity to transparently revoke anonymity can enhance security and promote fairness within networks. We suggest adaptations to this protocol. One can employ a decentralized group manager, where decisions and actions are executed upon consensus among a quorum of participants, potentially leveraging a consensus mechanism for governance.

We present two possible cross-chain protocols employing GS for anonymity assurance. In the first approach, a consortium of trusted notaries anonymously sign a *cctx*, resembling a Ring Signature protocol. It safeguards the operator's anonymity while also allowing for anonymity revocation in the event of suspicious activity on behalf of the operator. Alternatively, a user can sign a lock transaction on the source chain, blending in with a group of users interested in locking assets on a bridge. External entities can verify the locking of funds without uncovering the user's true identity. A trusted IM, acting as the group manager, privately executes the anonymity revocation algorithm to identify the user who locked the funds and mint tokens on the counterparty blockchain. The destination address could be passed through a confidential off-chain channel to the IM, such that only it could establish the link between transactions/addresses on both chains. If the IM deletes this link, unlinkability is guaranteed. Note that the source chain transaction's data could contain the destination address – but it would directly link those addresses. In the best-case scenario, users could achieve cross-chain pseudonymity using this final approach.

As of now, there are no cross-chain solutions based on group signatures. We believe this trade-off between privacy and accountability is worth exploring.

8. Listing of Vulnerabilities, Attacks, and Mitigations

Here, we provide further details on each vulnerability identified. Additionally, we summarize all mitigations in Table 6.2.

Honest Mining Assumption **($\mathcal{V}_1$).** This is an inherent vulnerability present in all networks that employ probabilistic finality, regardless of whether they serve as the foundational chains or as a relay chain facilitating interoperability. In networks with probabilistic finality, the consensus is not reached immediately, therefore, there is a period when transactions are not considered final and can be reverted. A single party that controls more than the security threshold of miners or validators, can authorize invalid *cctxs*. Additionally, updates to the core blockchain protocol can lead to soft or hard forks when a threshold of nodes does not adopt the changes. These might lead to the full reversion of transactions, or censorship [6]. In the worst-case scenario, BFT-based blockchains can completely halt under this scenario. Forks on a source chain may cause the smart contract on the destination chain to be unable to distinguish the main chain from the forks. It creates an opportunity for a *Double Spending Attack*, wherein an attacker can lock an asset X on the source chain and then mint X on both smart contract instances, one in each destination chain [58]. This vulnerability opens the door to various other attacks, such as *Sybil Attacks*, *51% Attacks*, or *Relay Poisoning Attacks*. As it is a fundamental assumption of permissionless blockchains with probabilistic finality consensus, one must always consider its implications. The literature proposes some mitigations, such as setting an appropriate security parameter based on the required number of confirmations ($\mathcal{M}_1$) [80], [90], which makes the probability of a block being reverted become negligible [173]. Also, one can insert maturity periods, i.e., time windows in which the block is not used for SPV ($\mathcal{M}_2$) [58]. To solve disputes, one can rely on blockchain views ($\mathcal{M}_3$) [55], [286]. As for a fork on the destination chain, potential solutions include explicitly specifying a destination chain where each fork would have a unique identifier ($\mathcal{M}_4$) [175] or synchronizing smart contracts from both instances of the destination chain through other cross-chain mechanisms ($\mathcal{M}_5$) [58].

Absence of Identity Verification ($\mathcal{V}_2$). A selfish miner (or colluding ones) can engage in a *Sybil Attack* where they control various identities in a network gaining more power than what is perceived, and thus increasing the level of centralization. For instance, invalid *cctxs* are accepted by the target chain just because a threshold of the network of validators is controlled by the same entity. Similarly, the absence of identity verification mechanisms also paves the way for *Collusion Attacks*, where apparent unrelated nodes collude to harm a bridge. Previous work showed that this vulnerability is impossible to mitigate without a centralized authority [287]. Hence, the first approach is a trusted authority that performs identity verification on the participants ($\mathcal{M}_8$) [96]. Other decentralized solutions require identity verification/mapping mechanisms using, for example, Self-Sovereign Identity ($\mathcal{M}_9$) [210] or making the creation of new identities expensive ($\mathcal{M}_{10}$). The latter is done usually through staking a high amount of funds when joining a network which is slashed in case of misbehavior [37], [58], [88]. The overall goal is to make the barrier to conduct such an attack higher than the potential profit. Single identities with higher stakes must be prioritized over splitting the stake between multiple identities, just like in PoS networks ($\mathcal{M}_{11}$). Note that a balance needs to be struck between these mechanisms and the adoption of a system because it might introduce a barrier to new users that do not want to stake so much funds [88]. Even though these mechanisms do not protect against *Sybil Attacks* performed by non-rational parties, performing this attack in networks with slashing mechanisms is very expensive and, therefore, unlikely.

Network Isolation ($\mathcal{V}_3$). In smaller networks, relayers can be intentionally isolated from the rest of the network during a period and misled to accept the attacker's chain as the longest [92]. These are called *Eclipse Attacks*. Actions according to this state are then issued against the target chain, violating the system's integrity. It is also present in centralized systems where an attacker with physical server access can interfere with its normal behaviour. Packets can be intercepted or dropped causing transactions to not settle in some networks [58], [79]. Additionally, some optimistic-based solutions [50] guarantee safety through challenges within predetermined time windows. If the attacker submits invalid block headers and isolates watchers for this duration, the target chain will eventually accept submitted block headers. The key to ensuring integrity is choosing an appropriate time window with an increased settlement time ($\mathcal{M}_6$), usually in the order of days or weeks. This introduces a tradeoff with usability. Additionally, to guarantee the liveness of the system, one can opt for the physical decentralization of infrastructure ($\mathcal{M}_7$).

Outdated Light Client State ($\mathcal{V}_4$). If source chain block headers are not relayed to the target chain, a light client may fall out of date, which can be due to an *Eclipse Attack* or the high costs of relaying information [58]. [69] calls *Liveness Attacks* to the ones where messages can be delayed. However, this compromises *liveness* of an interoperability solution and can cause unavailability or inaccurate transaction validation on the destination chain. Long-term solutions might encompass the usage of projects directed towards solving data availability problems, such as Celestia [171], which provides a modular approach to providing data that could be used for interoperability purposes ($\mathcal{M}_{16}$).

Wrong Main Chain Identification ($\mathcal{V}_5$). In a *Relay Poisoning Attack*, relayers can submit conflicting block headers to the target chain smart contract in an attempt to perform a chain reorganization in the source chain light client [6], [58], [92]. The problem surges when relayers submit valid source chain block headers (in terms of consensus rules) but with invalid transactions to the target chain which is tricked into accepting these blocks. If the hash rate of the submitting party is higher than the source chain's, the receiving smart contract has no way to distinguish which one is the real main chain, i.e., light client mechanisms must have ways to cope with *51% Attacks*, and *Long Range Attacks* on the source chain. However, if this is not the case, then it is possible to employ main chain identification mechanisms based on the consensus mechanism of the source chain ($\mathcal{M}_{18}$) – e.g., based on block difficulty in PoW or the number of block attestations in PoS. Note that the networks' consensus mechanisms are crucial here. Light clients of chains with instant finality will not suffer from this vulnerability or source chain forks [91].

Incorrect Event Verification ($\mathcal{V}_6$). As stated in Section 2.3, interoperability is driven by events emitted on both blockchains. In particular, in cross-chain bridges, the interoperability mechanism captures events emitted on the source chain and performs corresponding state changes on the destination chain. Correctly monitoring smart contracts and identifying events on the source chain is critical. Otherwise, one might be issuing transactions on the target chain without a corresponding event on the source chain, or the other way around [172]. The underlying causes range from the incorrect recognition of events emitted by malicious contracts [32], [173] or accepting events from tokens with similar names [174]. This vulnerability might originate *Event Forgery Attacks*. Solutions should listen to events from only whitelisted smart contracts ($\mathcal{M}_{12}$) [173], runtime monitoring modules ($\mathcal{M}_{13}$) [32], [174], or employing Distributed Signing Schemes where each party uses different monitoring strategies ($\mathcal{M}_{14}$) [82].

Acceptance of Invalid Consensus Proofs ($\mathcal{V}_7$). Malicious actors may attempt to construct invalid blocks, not adhering to the consensus rules, or include illegitimate transactions within valid blocks [130]. To address this, secure and up-to-date light client mechanisms promptly discard such block headers by verifying their consensus properties [58]. In Proof-of-Work (PoW) based clients, the block hash, current difficulty and difficulty adjustment policies are usually sufficient to detect these vulnerabilities. On the other hand, Proof-of-Stake (PoS) based light clients must keep track of epochs and sync committee information. Alternatively, in light clients of other consensus mechanisms, particularly where consensus participants are known a priori, corresponding keys must be updated in the smart contract ($\mathcal{M}_{15}$).

Absence of Chain Identification ($\mathcal{V}_8$). Cross-chain bridges built for Data Transfers, i.e., that allow arbitrary message passing, may enable interoperability between multiple chains [175]. One user might try to submit the same proof to multiple destination chains, to mint multiple representations of the locked token. Messages and proofs should maintain the source and destination chains' identifiers, to avoid integrity breaches in the form of *Replay Attacks*, or the submission of inclusion proofs from different and invalid contexts ($\mathcal{M}_4$).

Submission of Repeated Inclusion Proofs ($\mathcal{V}_9$). Attackers might try to repeatedly submit the same inclusion proof over and over again to try to prove a statement more than once [58], [76], [92], [130], [176] – i.e., a *Replay Attack*. For instance, after locking an asset X on the source chain, the attacker might present the corresponding Merkle proof multiple times to mint multiple representations of X on the destination chain. The same might happen the other way around when a user submits a *Burn* proof multiple times on the source chain to drain the escrow contract on the source chain (e.g., pNetwork hack in Table J). Bridge Sapling [92] solves this issue using a unique nonce generated in the locking phase when creating the zero-knowledge note commitment ($\mathcal{M}_{17}$). However, the solution is still vulnerable to collusion between the user and the relayer. The user can reuse the same input to the commitment note generator function ($\mathcal{V}_{41}$). XClaim [58] proposes the introduction of unique identifiers for each *lock* and *mint* transactions, synchronized between the target chain and the block headers submitted to the source chain light client. Rollups face the same challenge. Multiple transactions may be triggered on the L1, with only one *withdrawal* transaction initiated on the L2. As an example, this vulnerability was reported in the Polygon bridge in 2023 [250].

Counterfeiting Assets ($\mathcal{V}_{10}$). To guarantee safety, creating new assets on a destination chain through a cross-chain bridge must always be tied to events on the source chain. Failing to do so may lead to minting assets out of thin air [58], [92]. In a nutshell, the value of the assets in escrow on the source chain must not be lower than the value of all assets minted on the destination chain. A possible first mitigation strategy is having automatic liquidations of collateral once the risk factor decreases below a certain threshold ($\mathcal{M}_{19}$) [92], similar to DeFi lending protocols. Additionally, the protocol may employ mandatory staking making the misbehaving parties get slashed, yielding negative utility for those entities ($\mathcal{M}_{20}$) [58]. These solutions should be enforced by the overall protocol architecture – i.e., guarantee atomicity by design. It can be either enforced by decentralized watchers ($\mathcal{M}_{21}$) [50], [135], centralized systems ($\mathcal{M}_8$) [37], [73], [176], third-party networks with custom rules built-in to the consensus mechanism($\mathcal{M}_{22}$) [188], or cryptographic primitives set up jointly between users and operators ($\mathcal{M}_{23}$) [74], [98], [103], [105]. This vulnerability can also be inserted through a buggy smart contract implementation. In April 2022, Aurora Labs received a white hat bug report that could have minted more than $200M in unbacked assets through misuse of the *DelegateCall* solidity function due to preserving the original context of the caller [177].

Involuntary Timelock Expiry ($\mathcal{V}_{11}$). Due to the synchronous nature of some cross-chain protocols, if parties crash during a period may incur financial losses [38]. In HTLCs, if one party crashes and cannot provide the secret to the hashlock function until the timelock expires, the assets are returned to the other party. [80] relaxes the synchrony assumption using a witness network that maintains the state of the atomic swap, which is resilient to crashes from any party ($\mathcal{M}_{29}$). More work needs to be done to find a middle ground needs to be found between this and the last vulnerability, i.e., the chosen timelock should allow parties to redeem their assets within the specified period without being exposed to $\mathcal{V}_{13}$ ($\mathcal{M}_{30}$).

Unset Withdrawal Limits ($\mathcal{V}_{12}$). Cross-chain bridges, especially ones that rely on *lock-mint* patterns, maintain assets in escrow in the source chain. The sum of all these assets reaches billions of USD for most used bridges, which is naturally a honeypot for attackers. As we have been showcasing throughout this work, a simple bug in the business logic implemented in smart contracts can allow attackers to completely drain a bridge. Setting withdrawal limits for appropriate time windows is critical. These limits are tailored to specific bridges depending on the usual asset flow (e.g., hourly or daily) [175], [178]. Transactions get reverted, and bridge operators are warned and stop the bridge upon reaching a predefined threshold. Even though it is impossible to mitigate an attack with this measure, it is possible to maintain the eventual loss of funds limited to the defined value using withdrawal limits based on the usual flow of the bridge ($\mathcal{M}_{69}$).

Action Withhold / User starvation ($\mathcal{V}_{13}$). An attacker may intentionally abort a protocol execution, withhold an action to harm other parties or increase the possibility of profitability. For instance, when engaging in an asset exchange protocol using HTLCs, $\mathcal{E}_2$ might refuse to lock funds after $\mathcal{E}_1$ locked funds causing $\mathcal{E}_1$'s funds to be locked for the duration of the timelock. Additionally, both the initial and the refund transactions consume gas fees. Hence, in case of a revert, the victim still loses funds. These kinds of vulnerabilities are seen in *Griefing* or *Sore Loser Attacks* [40], [95], [179]. It can happen if the economic conditions (e.g., exchange rates) are suddenly not favourable to the attacker; or to mount a DoS attack on the counterparty [40]. The usual mitigation is the usage of *premiums* (cf. Section F.3) ($\mathcal{M}_{27}$) [40], [95].

Reference [106] proposes an adaption to HTLCs using *Attribute Verifiable Timed Commitments* where all parties can immediately abort a swap without waiting for the predefined timeouts ($\mathcal{M}_{29}$). Other mitigations are either using a trusted party to mediate the cross-chain swap ($\mathcal{M}_8$) [78] or an untrusted third party that engages in distributed signature scheme protocols ($\mathcal{M}_{23}$).

Unspecified Gas Limit ($\mathcal{V}_{14}$). A vulnerability [180] was found in Arbitrum. When withdrawing funds from the L2, the bridge contract on the L1 did not have the gas limit for the transaction set, and the user was not forced to set one and the contract on the L1 did ($\mathcal{M}_{65}$). Additionally, the transaction was marked as retryable – i.e., the relayer in charge would retry the transaction as many times as needed until running out of funds. This vulnerability can provoke *DoS*, *Eclipse*, or *Fund Draining Attacks* on bridge operators.

Resource Exhaustion ($\mathcal{V}_{15}$). Instead of inducing abnormal behaviour in the system, attackers may focus on disrupting the availability of a cross-chain solution. Centralized solutions are easily susceptible to Denial-of-Service (DoS) attacks. On the other hand, attackers might target single components in distributed networks. However, as the system becomes more decentralized, compromising the liveness of the entire bridge becomes increasingly challenging. We assess the vulnerability level based on whether a single component crash can compromise the correct functioning of the system. Additionally, we have already covered *Griefing* or *Sore Loser Attack*, where a user may be constantly starting a new swap without terminating the last one, leaving the other party with funds locked for the timelock duration. These are actually performing a DoS on the counterparty.

We go over some mitigations. Firstly, the protocol design should account for the possibility of DoS through the decentralization of components ($\mathcal{M}_{32}$) – e.g., leveraging a network of multiple relayers instead of only a couple of entities. Industry protocols such as CCIP, Axelar, and Wormhole follow this approach. Secondly, one must harden systems against DoS attacks employing rate-limiting strategies or, if applicable, challenge-response tests ($\mathcal{M}_{50}$). Ultimately, components that are suitable for on-chain design and implementation (as contracts) should be developed as such, such that executing a Denial-of-Service (DoS) attack on that component necessitates a DoS attack on the entire network ($\mathcal{M}_{49}$) [58].

Single Point of Failure ($\mathcal{V}_{16}$). Single points of failure can be analysed in multiple dimensions. Firstly, at the infrastructure level, a protocol centralized on a single machine is subject to failures and might compromise the liveness of the solution. Additionally, there might be a single point of failure from an architectural perspective, where one application crashes if one entity (that might control several pieces of physical infrastructure) is compromised. Centralized relayers [79] or operators [187] are vulnerable. The most common mitigation is decentralization both physically ($\mathcal{M}_7$), architecturally ($\mathcal{M}_{32}$) and operationally ($\mathcal{M}_{47}$). By relying on a decentralized network, denying service to one component requires denying service to all replicas of that component [58], [79].

Publicly Identifiable Operators ($\mathcal{V}_{17}$). Solutions where operators are public and identifiable are vulnerable to multiple attack vectors – *Bribery*, *Collusion*, *DoS*, and *Phishing* are some examples. These can compromise both the safety and liveness of a cross-chain bridge. Bool Network [43] puts a step forward by introducing an interoperability mechanism composed by an evolving committee that changes every epoch ($\mathcal{M}_{44}$). Each member is also hidden among a ring based on a *Ring VRF*, making linking one member to a digital signature impossible ($\mathcal{M}_{45}$). It not only protects the privacy of operators, as well as their security given that these mechanisms act as a barrier for attackers. Moreover, the usual cybersecurity practices applied in web2 should also be followed ($\mathcal{M}_{46}$).

Misaligned Incentive Mechanisms ($\mathcal{V}_{18}$). Incentivization is paramount in decentralized systems. In the BAR behaviour model [183], Rational players follow strategies that maximize their profits – i.e., they might choose to deviate from the protocol rather than following the rules due to the more attractive economic incentives. *Collusion* and *Bribery Attacks* are some examples of attacks.

In *Asset* or *Data Transfers*, centralized components such as TEEs [77], or decentralized relayers can collude, for example, to inject forged Merkle Tree Roots into the destination chain [82]. Additionally, one can intercept and drop packets from other relayers. In this latter case, instead of adding invalid transactions to new blocks, relayers can submit only a subset of updates to induce an inconsistent state in the target chain. In optimistic fraud-proof-based solutions, watchers can collude and not dispute any illegal headers for the duration of the time window [50].

In *Bribery Attacks* [184], the briber and bribee can profit from withholding the announcement of new blocks until the briber successfully mines and announces the new block to a network. If the number of relayers is small enough, an attacker can bribe them to relay block headers which allows the attacker to double spend funds. This idea is similar to censorship attacks, where miners are incentivized to ignore transactions from certain addresses [185].

There are multiple mitigation strategies proposed in academia. Firstly, there is a need to study the best way to align incentive mechanisms to make these attacks less probable and less profitable. Some examples leverage game-theory principles applied to asset exchanges [138] or asset transfers [77] ($\mathcal{M}_{31}$). For more centralized solutions, where collusion can happen between fewer parties, but usually each has more power, decentralization is an option ($\mathcal{M}_{32}$) [186], introducing more and different parties to make the attack more expensive ($\mathcal{M}_{33}$) – as the number of participants increases, a higher number also needs to be convinced to follow the attack [50]. We add that the mining power needs to be distributed accordingly to that growth. On the other hand, employing regulatory authorities that ensure parties are accountable for their actions might be a solution ($\mathcal{M}_{8}$) [70]. Another area of research has been the exchange of assets co-owned by multiple parties, which are vulnerable to collusion [103]. Some protocols engage in distributed signature schemes between users and operators ($\mathcal{M}_{23}$). MAD-HTLC [99] presents an interesting solution. It shows how to secure HTLCs using MEV ($\mathcal{M}_{34}$). By choosing which transactions go into which blocks, miners can accept transactions that yield more profit than the "normal" transaction selection process. The protocol ensures that if one party misbehaves, both lose their assets (the *mutually assured destruction* principle). The main limitation of this work is assuming all blockchain miners are rational and always follow the most profitable path, which might not be totally realistic.

We believe more research on the analysis of the strategy followed by rational players who maximize profit in cross-chain is needed, which must guarantee strong Nash equilibrium [39].

Token Price Volatility ($\mathcal{V}_{19}$). Cryptocurrency-based bridges (AE, AT) also suffer from the high volatility in token prices which can lead to unfair trades [43], [92]. Decentralized Finance protocols inherit this vulnerability, which cannot be entirely eliminated. In HTLCs, if a sudden cryptocurrency devaluation occurs, one party might cease participation midway because the initial conditions that were agreed upon are no longer valid [95]. Lilac [97] proposes each party to lock assets in parallel which reduces the duration of the swap ($\mathcal{M}_{35}$) and consequently reduces the time window for users to observe price fluctuations ($\mathcal{M}_{36}$). However, it is still vulnerable to *Collusion* and *Sore Loser Attacks*.

For bridge-based protocols, we find that the security of a bridge is highly dependent on the valuation of the assets in escrow, which, if not secured, can cause the issuance of unbacked tokens, or trigger massive liquidations [92]. Consider an asset X that is used as collateral to mint an asset Y on the destination chain. If suddenly X loses valuation, then Y might become uncollateralized – i.e., the new value of X does not cover the value of Y. XClaim [58] proposes three solutions: 1) over-collateralization to account for some slippage ($\mathcal{M}_{37}$), 2) enable the adjustment of the amount locked according to the updated exchange rates ($\mathcal{M}_{38}$), or 3) introduce automatic liquidations so that it becomes impossible to have the locking party getting uncollateralized ($\mathcal{M}_{39}$). XCC [98] points out that over-collateralization is not scalable and not attractive to vaults. Therefore, the assets are in a timelock jointly controlled by the user and the vault, and these funds are only transferred to the vault in some checkpoint periods when the commitment needs to be renewed.

Multi-chain networks can also suffer from this vulnerability. Platforms, where the token that guarantees economic security is endogenous to the network, might see their economic security decrease because of a sudden devaluation [124].

Centralized Power ($\mathcal{V}_{20}$). Centralization must be evaluated across different layers. Protocols can rely on centralized infrastructure [79] or on distributed infrastructure but are still mainly controlled by a single entity [187]. As an example, protocols might have a decentralized architecture but rely on centralized governance procedures [188] or centralized computation conducted by L2 bridge operators [66], [189]. Possible consequences are liveness compromise, transaction censorship or transaction reordering. The first problem we identify is liveness compromise. If a centralized system crashes or halts temporarily, the whole protocol will stall for the same duration. Even though this might be irrelevant for some protocols, it is crucial for time-sensitive ones that rely on performing actions within a certain amount of time [185]. The most straightforward solution in these cases is decentralization ($\mathcal{M}_{32}$) [48], [186]. Moreover, protocols can be the target of *Censorship Attacks*. In such attacks, the miners of one blockchain may not insert a transaction in a block [36], rollup operators may exclude L2 transactions from batches submitted to the L1 [108], [187], or relayers choose to drop *cctxs* instead of forwarding them to the target chain [82]. From the BAR classification of actors, the motivation to engage in such an attack may vary. On the one hand, the attacker might want to harm the system by censoring at their own will, behaving as Byzantine. On the other end of the spectrum, attackers might be rational and choose a path that yields external gains, such as leveraging (cross-chain) MEV opportunities. Nevertheless, the ability to censor might also have some benefits. Blacklisting functionalities are censorship actions that protect the protocol against pre-known users/contracts associated with some form of illegal activity (e.g., having used the US-sanctioned protocol Tornado Cash). Instead of relying on centralized architectures, protocols can have defined governance procedures to implement these changes. However, we acknowledge that there have also been several governance attacks [192], [288]–[290]. In rollups, usually, there is always a way to bypass censorship performed by sequencers, by issuing transactions directly to the Layer-1 contract [190] as done in StarkEx and ZkSync – i.e., multiple components have overlapping capabilities ($\mathcal{M}_{43}$).

Verifier's Dilemma ($\mathcal{V}_{21}$). The Verifier's Dilemma, initially proposed by [191], shows that rational blockchain miners benefit from skipping the verification of blocks to gain an advantage in proposing subsequent blocks. The probability of such behaviour increases when blocks contain computationally expensive transactions. We acknowledge that cross-chain solutions based on third-party networks also suffer from this vulnerability, where a *cctx* might not be fully validated. A possible solution is parallelizing the verification of non-conflicting transactions within the same block to decrease verification time ($\mathcal{M}_{24}$), or dividing computational-heavy transactions into multiple blocks ($\mathcal{M}_{25}$). However, it is not clear how to guarantee atomicity under the latter scheme. Alternatively, one can assign entities with different responsibilities, separating the verification from the computation logic ($\mathcal{M}_{26}$) [211].

Manipulation of Exchange Rates ($\mathcal{V}_{22}$). Token prices from external sources are written into blockchains by oracles. Oracles can be manipulated to send an erroneous price feed [192]–[195]. Bridges can therefore use the wrong exchange rates leading to unfair or unrealistic trades. In the worst-case scenario, a protocol may fail to maintain proper collateralization of the assets issued. Furthermore, similarly to $\mathcal{V}_{19}$, during the lock period in HTLCs, one party might either manipulate oracle data or influence the market by valuing or devaluing a token after the first party locked assets. We call these *Exchange Rate Poisoning Attacks*. In April 2023, Allbridge [291] suffered a ~500k USD hack after a liquidity pool's swap price manipulation [22]. Miners who exploit MEV opportunities can also front-run transactions to profit from these (de)valuations. The usual mitigation to address oracle manipulation is to base transactions on multiple sources of data, not single oracles, nor controlled by the same entity ($\mathcal{M}_{40}$) [186]. Multiple works have studied the consequences and mitigations for the fluctuations caused by MEV ($\mathcal{M}_{41}$). Some examples are confidential transactions [213], or employing confidential mempools [212]. We refer the reader to [189], [192], [292]–[298], for more information on MEV. Liquidity pool-based bridges can also suffer from manipulation of swap prices.

Unfair Transaction/Event Ordering ($\mathcal{V}_{23}$). Transaction ordering techniques enforced by blockchain miners through MEV are also found in a cross-chain scenario [82]. Similarly to the unfair ordering in the miners' mempool, the interoperability mechanisms that relay block headers, events, or any other type of proofs between blockchains, can also be subject to custom order based on the maximum extractable profit. Just like blockchain miners, consider a decentralized network of relayers that relay *cctxs*, where the default ordering strategy is based on the received fees. If relayers are incentivized to choose one *cctx* over another some fairness issues arise. The same happens in protocols based on synchronous communication between parties (e.g., HTLCs), that might have their transactions stalled, and, in the worst-case scenario, enter a block after the defined timeout. The most straightforward solutions are guaranteeing the confidentiality of *cctxs*, in a way an adversary cannot peek at other users' transactions ($\mathcal{M}_{41}$), or enforcing a predefined transaction ordering policy ($\mathcal{M}_{42}$) [82], [213].

Lack of Access Control ($\mathcal{V}_{24}$). With the rapid evolution of decentralized applications' development, the complexity of such apps has increased exponentially. Inherently, components responsible for managing locked funds or creating new assets become prime targets for hackers. Consequently, it is crucial to enforce robust access control policies to manage access to these contracts effectively. Unauthorized access to critical smart contracts causing the transfer of tokens to attacker-controlled addresses or whitelisting an attacker-controlled address as valid escrows have been the cause of cross-chain hacks (cf. Section 6.3). Some examples are Qubit, Meter, Thorchain, Nomad, Poly, and BNB, where the attackers could replace whitelisted relayers' keys and execute smart contracts with arbitrary business logic [196]–[201]. We present two mitigations. Firstly, contracts should go through multiple iterations of audits before being deployed on a chain ($\mathcal{M}_{51}$). Also, considering the requirement for continuous code upgrades, one must plan and perform a coherent lifecycle of audits. Ultimately, we reckon that the recurrence of similar attacks is primarily attributable to the absence of a standardized architecture and design pattern for developers to construct cross-chain bridges while ensuring robust access control ($\mathcal{M}_{52}$). For now, the bridge design remains an ad-hoc approach, with each bridge operating on a distinct architecture, making it susceptible to errors and hindering technological advancement.

Conceed Approvals to Third Parties ($\mathcal{V}_{25}$). The usage of functions such as *approve()*, *permit()* and *transferFrom()* available in some token standards such as ERC20, open the door to novel attacks [173]. The baDAPProve problem, found in the Multichain bridge hack [203], refers to the users permitting the bridge contract to spend tokens on their behalf, which allows them to save gas fees. The main problem surged when the bridge contract was compromised, causing the attacker to drain funds directly from user accounts. The evident mitigation is not issuing approvals for more funds than what is strictly necessary ($\mathcal{M}_{53}$).

Outdated third-party library version ($\mathcal{V}_{26}$). Vulnerability X warns against automatic library upgrades to prevent unintended code behaviour. However, infrequent updates may leave security patches unapplied [204]. We emphasize the importance of reviewing packages and making sure they were subject to recent audits ($\mathcal{M}_{78}$).

Unsafe Third Party Modules ($\mathcal{V}_{27}$). As usual in software development, code relies on third-party modules or libraries. These libraries can insert vulnerabilities into the codebase, which may weaken the source code [172], [175], [181], [182]. For example, the BNB bridge was hacked due to a bug in the digital signature verification method outsourced to a vulnerable third-party module [201]. Ensuring third-party libraries are bug-free is critical, therefore, check how long ago that code was reviewed by a specialized team. All audit reports reviewed assume that external libraries are correct, which seems acceptable due to the limited scope one has. However, teams should review previous audit reports performed by other companies ($\mathcal{M}_{78}$). Another usual mitigation is the absence of library version auto-upgrades ($\mathcal{M}_{58}$), which might unwillingly introduce breaking changes in the code.

Dead Code ($\mathcal{V}_{28}$). Codebases are consistently being updated and upgraded as knowledge evolves. A noteworthy vulnerability behind, at least, the Qubit and Multichain hack, is the presence of dead code within the deployed smart contracts which allowed attackers to execute malicious operations (cf. Table J). IDEs usually have linting tools embedded that allow for identifying unused functions and raising warnings to users ($\mathcal{M}_{59}$).

Usage of non-standard/conventional naming ($\mathcal{V}_{29}$). Different programming languages use specific naming conventions for the names of variables, functions [205], or the usage of curly brackets. The developer should be aware of the best practices to code in the respective languages and follow them flawlessly to help developers, clients, and auditors ($\mathcal{M}_{79}$).

Inconsistent smart contract engine version ($\mathcal{V}_{30}$). Multiple audits have found that, within the same project, smart contracts are using different versions of smart contract engines [175], [182], [205]. Versions differ in their functionalities. Therefore, to maintain the codebase coherent, applying the same version across different files is advisable to avoid incompatibilities and maintain standardization ($\mathcal{M}_{80}$).

Unconventional code/testing architecture ($\mathcal{V}_{31}$). Unconventional architectures at the protocol and implementation levels present a challenge to building secure and scalable bridges. At the implementation level, it is difficult for auditors to evaluate the codebase and for practitioners to understand the different components' locations. Additionally, test-wise, having uncommon test structures or architectures makes it harder to understand the testing methods and the functionality under evaluation [181], [204]. Even though static analysis tools can perform code coverage analysis, understandability is the most crucial factor for a robust codebase ($\mathcal{M}_{81}$).

Reentrancy ($\mathcal{V}_{32}$). *Reentrancy Attacks* are very common in smart contracts (still happening in 2023 [239]). The overall idea is that a smart contract calls an untrusted contract, and the latter recursively calls the initial one to manipulate its internal state. This vulnerability was found in one reviewed audit [175]. The most common mitigation for this vulnerability is to update the internal state of a contract before making an external call to another contract ($\mathcal{M}_{82}$).

No emission of events upon critical state changes ($\mathcal{V}_{33}$). Cross-chain systems revolve around events. Off-chain mechanisms listen for events that indicate state changes and sometimes forward them to other chains. Not emitting events upon state changes can have serious consequences [172], [206]. Firstly, it might jeopardize the integrity of the bridge. Secondly, off-chain monitoring mechanisms cannot understand what is happening within the smart contract. Finally, it hardens the job of debugging and auditing the code by third parties. Worse than not emitting events is emitting events with wrong data [182], which may cause an attacker to drain the bridge. The emission of events is crucial and should be assured to guarantee the integrity of the bridge ($\mathcal{M}_{83}$).

Inconsistent bridge contract interfaces ($\mathcal{V}_{34}$). Bridges are composed of multiple components that must communicate with one another through standardized interfaces [207]. Not guaranteeing consistent bridge contract interfaces may cause an indefinite loss of funds, such that messages sent by one party are not understood by the other. We emphasize the need for standardizing code architectures, such that the same code package is shared between multiple components instead of duplicating code ($\mathcal{M}_{84}$).

Out of order transaction execution ($\mathcal{V}_{35}$). An auditability to Arbitrum's code has found a vulnerability where an attacker can exploit the absence of an ordering mechanism to deny a user access to its assets [172]. The vulnerability exploits the way retryable tickets are implemented in the bridge. Mechanisms that guarantee the atomicity and transaction ordering from the L1 to the L2 are still to be implemented ($\mathcal{M}_{85}$).

Absence of storage gaps for upgradeable smart contracts ($\mathcal{V}_{36}$). A recent recommendation for the development of upgradeable smart contracts is the addition of storage gaps to allow for additional storage variables in the future ($\mathcal{M}_{86}$). The main advantage of using this pattern is to allow for inserting new state variables in the future without compromising the storage compatibility with existing deployments [208].

Integer overflow and underflow ($\mathcal{V}_{37}$). Attempting to store values higher or lower than the largest and least value supported by a data type incurs an overflow or underflow, respectively. This vulnerability might allow an attacker to drain a bridge by convincing the bridge that the value is within the expected range when it is not. Static analysis tools suffice to mitigate this vulnerability ($\mathcal{M}_{87}$).

Absence of Sanity Checks ($\mathcal{V}_{38}$). Throughout the codebase, there must be checks to ensure the bridge functions as intended, safeguarding its integrity. This encompasses validating inputs (e.g., make sure an address is either an EOA address or an authorized contract address with a predetermined function [172], [175], [178]), function return types [182], operations for arithmetic errors [204], ensuring there are no operations on null addresses [172], [178], [181], inconsistent data type conversions [181], [182], and the size of the payload being transferred in the bridge [204]. There may also be application-specific checks to guarantee that addresses provided as input are not part of critical infrastructure – e.g., an attacker might provide an infrastructure contract as input to attempt an attack – or make sure that there is no possibility of having the same validator registered in the bridge contract twice [205]. Multiple static analysis and runtime analysis tools might help mitigate and lower the incidents related to these vulnerabilities ($\mathcal{M}_{87}$).

Mismatch between code and comments/documentation ($\mathcal{V}_{39}$). Several audits have revealed occasional inconsistencies between the code and its accompanying comments [175], [205], [206] or documentation [181], [182], [204], [205]. We acknowledge the possibility of human error during the implementation phase. However, such discrepancies should be diligently prevented, as they can mislead both developers and auditors. Solutions range from running checks once pull requests are about to be accepted to the usage of AI tools to detect inconsistencies ($\mathcal{M}_{88}$).

Uninitialized variables ($\mathcal{V}_{40}$). In September 2022, a white hat found a massive vulnerability in the Arbitrum bridge [209]. When analysing the source code, the hacker noticed that there was an address variable uninitialized, which was purposely wiped after initialization (through an upgrade function) to save gas fees. Any user could call the *initialize()* function passing a controlled address, which would serve as the new escrow contract. To mitigate these exploits based on this vulnerability, we should take security as paramount, and optimizations as the next step ($\mathcal{M}_{66}$). We believe the industry is not at that stage yet. A similar vulnerability was found in the Wormhole bridge contract with an uninitialized proxy [299]. The white hackers received 500k and 10M USD respectively for reporting these bugs through bug bounty programs.

Leakage of ZK private inputs ($\mathcal{V}_{41}$). As introduced in Section 4.4.3, the CRS used to create and verify ZK proofs is computed using private inputs provided to an MPC scheme. The leakage of these inputs can lead to an adversary being able to forge proofs and generate cross-chain state transitions that violate the defined cross-chain rules. This vulnerability has not been observed in a hack. However, it is crucial to guarantee integrity in ZK-based bridges. Once used, private inputs must be immediately disposed of ($\mathcal{M}_{67}$) [44].

Other Smart Contract Vulnerabilities ($\mathcal{V}_{42}$). We do not explore all smart contract-related vulnerabilities due to their extension. Rather, we point the reader to an extensive work surveying vulnerabilities in this context [215]. Nonetheless, we present some bridge-related vulnerabilities found. These range from signature verification bypass in the Wormhole hack [205], incorrect usage of modifiers [172], [182], unauthorized smart contract calls in the first PolyBridge hack [200] and wrong function visibilities [182]. Another significant concern is the existence of non-reverting fallback functions. These functions are invoked when a contract receives a function call that does not correspond to any defined function. The absence of proper revert statements in these fallback functions allows invalid transactions to succeed, resulting in inconsistent state changes. We refer to all smart contract vulnerabilities' mitigations as $\mathcal{M}_{54}$ [215]. We outline three mitigations, namely conducting thorough code reviews ($\mathcal{M}_{55}$), implementing rigorous testing ($\mathcal{M}_{56}$), and regularly conducting security audits ($\mathcal{M}_{51}$).

Inadequate Key Management ($\mathcal{V}_{43}$). The compromise of cryptographic keys is one of the main sources of hacks in cross-chain bridges [110], [173]. Even worse than compromising a single key, is compromising multiple keys, which can cause multi-signature account hijacking [82]. In the Ronin bridge hack, an entity with access to several keys was compromised, allowing the attackers to control the system. In the Harmony Bridge hack, the attacker exploited two keys in a 2 out of 5 multi-sig. In Anyswap's hack in 2021, there were two signatures generated using the same random value for the ECDSA signature generation algorithm, which allowed retrieving the underlying private key, due to a primitive implementation. RFC 6979 [300] already presented a solution to this problem. The BXH hack happened because the attacker accessed the administrator's private key through a phishing attack. In July 2023, the Poly Network was hacked for the second time due to a 3 out of 4 multi-signature compromise (not represented in Table 5). Projects need to improve key management strategies, for example, with the usage of hardware wallets ($\mathcal{M}_{60}$) [216]. Other mitigations are the increase of the number of validators and thresholds in multi-signature wallets ($\mathcal{M}_{61}$), and decentralization ($\mathcal{M}_{47}$), where one entity should not have access to multiple cryptographic keys. Additionally, keys can be protected with additional authentication procedures, be it symmetric keys, or passwords ($\mathcal{M}_{62}$) [66].

Physical Infrastructure Backdoors ($\mathcal{V}_{44}$). Infrastructure backdoors create numerous potential attack vectors. However, assessing the extent of harm to the system proves challenging, as it relies on the level of control the compromised component wields over the system, primarily influenced by factors such as decentralization and deployment methods (e.g., on-premises or cloud infrastructure). Nevertheless, remote blockchain nodes can be reachable by RPC or HTTP ports which can be used to transmit malicious transactions or perform DDoS attacks [66]. Firewalls should be set up to only accept connections from identified IP addresses ($\mathcal{M}_{63}$), or the use of symmetric keys to authenticate requests ($\mathcal{M}_{64}$) [66]. Usual mitigations for infrastructure backdoors should also be applied here ($\mathcal{M}_{46}$).

Social Engineering-related Vulnerabilities ($\mathcal{V}_{45}$). Attacks such as *Phishing* or *Ransomware Attacks* can be performed through social engineering practices, usually in social media or untrusted websites. Vulnerable files or hyperlinks with attractive messages are disseminated through these platforms or email [202]. Such attacks have targeted cross-chain solutions multiple times, the last being in March 2023 [214]. The usual mitigations for this type of vulnerability are applied here, more related to increasing the awareness of actors for these attacks ($\mathcal{M}_{77}$).

9. Privacy Leaks of Interoperable Systems and its Mitigations

In this section, we provide insights on the main privacy leaks of interoperability systems and how to mitigate them.

Zero-Knowledge-based IMs ($\mathcal{L}_1$). In systems using zero-knowledge proofs for interoperability, there may be a trusted ceremony (for some proof systems like Groth16 [301]). In those, a common reference string is shared to create the proving and verification keys. Depending on the input information, one could potentially reveal confidential information about the participants or the transactions being conducted, violating privacy guarantees. For example, a user provides their address or private key as input to the ceremony. A mitigation includes providing a secure random string as input to the trusted ceremony, such as a UUID v4.

Inference Attacks ($\mathcal{L}_2$). Zclaim [92] mentions the possibility of parties inferencing the identity of users through the amounts in *lock* and *release* transactions in which they are involved. Proposed mitigations are using a splitting strategy [92], where the amounts to be bridged are split into several transactions, and using ZK proofs [156] to prove there is a correct transaction without disclosing the specific transaction. Other work [89] proposes a virtual address generator function $\varphi(x)$ so that the real sender address is not disclosed outside the blockchain. However, these simple mitigations can be countered by employing relatively simple heuristics (for example aggregating information about certain user transactions, namely the destination address and amounts).

Common Secret Deployment ($\mathcal{L}_3$). In HTLCs, the hash of the secret generated by Alice is published both in the source chain and in the target, by Alice and Bob respectively. Therefore, local transactions in different blockchains can be linked together as belonging to the same *cctx*, which breaks *unlinkability*. [100] presents a novel protocol for atomic swaps by utilizing the Diffie-Hellman key exchange (an open source implementation available here [285]). The protocol aims to establish a shared key without publishing it on the blockchains. In addition to this key exchange, the authors propose an algorithm that employs Adaptor Signatures. Similarly, Cai et al. [101] use Parllier homomorphic encryption, leveraging its additive homomorphism property. The protocol involves establishing a shared secret through off-chain agreement and performing computations on this shared secret. One of the main advantages of solutions based on Timed Commitments or Timed Signatures covered in Section 4.4.4 is that no timelock is deployed on-chain, which preserves the fungibility of transactions [42] – i.e., it is not possible to distinguish transactions from an atomic swap protocol and normal intra-blockchain transactions, which helps to guarantee *cctx* unlinkability.

Internal Privacy Leaks $\mathcal{L}_4$. Privacy leaks may be voluntary or involuntary, and those include revealing, for example, mappings between addresses and real-world identities (which the majority of regulated cryptocurrency exchanges have); mappings between two related on-chain accounts [156], or simply operational information (for example, one can map the provided Endpoint contracts with the LayerZero operator [302]. This mapping allows discovering what is the revenue by addresses probably controlled by LayerZero).

User-Generated Privacy Leak ($\mathcal{L}_5$). In multiple situations, users leverage insecure practices in privacy-preserving applications or platforms, such as mixing services (e.g., Tornado Cash) or privacy-preserving blockchains (e.g., Monero). It has been shown by multiple previous works that the privacy level offered by these solutions can be jeopardized by multiple factors [141]–[145].

10. Real World Cross-Chain Bridge Hacks

We provide further details on past cross-chain bridge hacks. Table J presents the date and amount of hacks that account for more than 3 billion USD. Additionally, we describe in detail each analyzed hack and propose a set of mitigations in table 5.

TABLE 8. DATASET OF CROSS-CHAIN BRIDGE HACKS ORDERED BY DATE

Bridge Name	Hack Date	Amount (Million USD)
Thorchain	June 2021	0.14
Thorchain	July 2021	5.00
Thorchain	July 2021	8.00
Thorchain	July 2021	0.08
Chainswap	July 2021	4.40
Chainswap	July 2021	0.80
Anyswap	July 2021	8.00
Poly Network	July 2021	4.40
Poly Network	August 2021	611.00
pNetwork	September 2021	13.00
BXH	October 2021	139.00
Nerve	November 2021	0.54
Multichain	January 2022	3.00
Qubit	January 2022	80.00
Wormhole	February 2022	326.00
Meter	February 2022	7.70
Ronin	March 2022	624.00
Harmony	June 2022	100.00
Nomad	August 2022	190.00
CelerNetwork	August 2022	0.24
BNB	October 2022	566.00
QANplatform	October 2022	2.00
Rubic	November 2022	1.20
pNetwork	November 2022	10.80
Rubic	December 2022	1.40
Multichain	February 2023	0.13
Allbridge	April 2023	0.57
Cellframe Network	June 2023	0.07
Multichain	July 2023	130.00
Mixin Network	September 2023	200.00
Heco Bridge	November 2023	99.00
Orbit Bridge	December 2023	81.88
Socket	January 2024	3.30
	Total	**3221.65**

TABLE 9. DESCRIPTION AND POSSIBLE MITIGATIONS FOR SOME OF THE MOST PROFITABLE CROSS-CHAIN BRIDGE HACKS.

Bridge & Refs	Mapping to our model?	Description	Mitigations
Ronin Bridge [303], [304]	The validators were compromised. The attackers compromised 5 out of 9 validators – the exact threshold.	• Nobody noticed for 6 days. No monitoring existed. • 5 out of 9 validators were needed to approve transactions, whereas 4 were controlled by the same entity Sky Mavis. • The Axie DAO controlled the 5th validator, however, there was a gas-free RPC node through which Sky Mavis had access to the Axie DAO validator.	• $\mathcal{M}_{68}$ – Insert monitoring procedures in the bridge. • $\mathcal{M}_{60}$ – Improve cryptographic key management (e.g., cold wallets, or multi-signatures). • $\mathcal{M}_{61}$ – Increase the number of validators, and the threshold necessary to deem a proof valid. • $\mathcal{M}_{57}$ – Audit not only smart contracts but all the infrastructure that is behind. • $\mathcal{M}_{69}$ – Set withdrawal limits. • $\mathcal{M}_{70}$ – Do not give excessive permission to individual external entities.
PolyBridge [200], [305]	The contract that manages the public keys of active keepers. The user accessed this contract through the bridge one.	• EthCrossChainManager (the bridge contract) has a function executeCrossChainTx() that calls a destination smart contract to unlock tokens passed as an argument. Also, there is a EthCrossChainData contract that manages validators' public keys. • The attacker accessed the function putCurEpochConPubKeyBytes(bytes) that updates the validators' public keys in the EthCrossChainData contract. • Since EthCrossChainManager is the owner of EthCrossChainData, the call would pass the isOwner() check. • Also, to trick the EthCrossChainManager contract to call that function, the attacker relied on his knowledge of the EVM. He found a hash collision where the first four bytes would match the first four bytes of the signature hash of the target function.	• $\mathcal{M}_{52}$ – Smart contracts accessed by users should not have direct access to management smart contracts. • $\mathcal{M}_{71}$ – In dynamic bridges, when receiving a contract address as an argument, check that it represents a contract and, if possible, that the corresponding method being called is valid. • $\mathcal{M}_{69}$ – Set withdrawal limits.
PolyBridge [306]–[309]	Validators' keys were compromised	• 3 out of 4 keys were compromised for a 3 out of 4 multi-sig. • The attacker authorized withdrawals in multiple destination chains	• $\mathcal{M}_{60}$ – Improve cryptographic key management (e.g., cold wallets, or multi-signatures). • $\mathcal{M}_{61}$ – Increase the number of validators, and the threshold necessary to deem a proof valid.
BNB Bridge [201], [310]	Buggy proof verification mechanism.	• The Cosmos' IAVL proof.go implementation had a bug when checking the validity of the proof on the target chain.	• $\mathcal{M}_{51}$ – Make sure third party components/libraries have been audited by multiple entities. • $\mathcal{M}_{68}$ – Freeze deposits and withdrawals to and from the bridge.
Wormhole [311], [312]	A bug was introduced in the proof verification component in the target chain.	• The bridge relies on a set of 19 guardians. • Guardians sign events emitted on both chains. • Due to a mistake, the version solana program being used didn't verify correctly the signatures. • The attack happened a few hours after a patch was fixed in GitHub.	• $\mathcal{M}_{72}$ – Do not publish (push) critical fixes before those changes are deployed. • $\mathcal{M}_{58}$ – Perform security audits with the different versions of the libraries being used. • $\mathcal{M}_{69}$ – More than 93,000 ETH was moved back to Ethereum which could have been avoided if transfers were paused/blocked
Nomad Bridge [198], [313], [314]	The verification of the lock proof would deem every message valid.	• The contract was initialized with 0x0 as a trusted root. • If calling directly the *process()* function in the Replica contract (the bridge smart contract in each chain), it would internally call the *acceptableRoot()* function with the result of *messages[_messageHash]*. • If a message was invalid, i.e., was not yet proved, *messages[_messageHash]* would return 0x0, calling the *acceptableRoot()* function with 0x0 as a parameter. Given that 0x0 was set as a trusted root, the validation would be successful.	• $\mathcal{M}_{73}$ – A bug was introduced shortly after an audit by an external team. Changes in critical components of the code should not be done after an audit.
Harmony [315]–[317]	Validators' keys were compromised	• The bridge relied on a 2 out of 5 multi-signature and two addresses in a hot wallet were compromised. • Harmony stopped the Horizon bridge to prevent further transactions, however, the attacker was able to swap funds to ETH in the Ethereum blockchain.	• $\mathcal{M}_{60}$ – Improve cryptographic key management (e.g., cold wallets, or multi-signatures). • $\mathcal{M}_{61}$ – Increase the number of validators, and the threshold necessary to deem a proof valid.
Qubit Finance [196], [318]	A deprecated function allowed minting tokens without a valid proof.	• Developers forgot to remove a deprecated function from the smart contract that allowed a zero address to call a *safeTransferFrom()*. • The attacker called the deposit function in the source chain (Ethereum) passing an invalid token. The transaction did not revert and the smart contract in the destination chain was instructed to mint xETH tokens to the attacker's address on BSC.	• $\mathcal{M}_{59}$ – Automatic tools allow identifying deprecated (or unused) functions. • $\mathcal{M}_{55}$ – Code reviews should suffice to mitigate this vulnerability.
Meter [197]	Implementation bug in the deposit function in the contract deployed to the source chain.	• The implementation was a modified version of Chainsafe's ChainBridge. They added a new function to deal with native tokens. • The attacker exploited the existing *deposit()* function. The handler (called from the *deposit()* function) had a faulty condition to check for wrapped assets which assumed tokens were already transferred by the bridge. The attacker could pass an arbitrary amount of funds that were transferred to the attacker's address.	• $\mathcal{M}_{74}$ – Know and understand thoroughly a codebase before forking it, especially before creating new functionality. • $\mathcal{M}_{51}$ – Audit code before and after changing code.
Thorchain #1 [199]	Locking a token with a name similar to ETH was interpreted as valid ETH.	• Logical bug in the Ethereum Bifröst smart contract caused fake tokens named "ETH" to be interpreted as real ETH.	• $\mathcal{M}_{12}$ – Whitelisting valid token contracts.
Thorchain #2 [319], [320]	Implementation bug in the relayer. The attacker pretended to transfer funds, taking advantage of the deposit verification mechanism.	• The hacker wrapped the router with a specific smart contract implementation that tricked the bridge smart contract. • They were able to trick the bridge smart contract into processing *msg.value* tokens, even if 0 was provided. • The vulnerability was already known (there was also a comment in the code pointing to the affected loop)	• $\mathcal{M}_{75}$ – Fix bugs as soon as they are detected. • $\mathcal{M}_{68}$ – This attack was performed using multiple transfers, thus, a limit could have been set to limit the number of transfers – e.g., per user, per day, or both.
Thorchain #3 [321]–[323]	The user forged an event deposit. Convinced the bridge that a certain action took place.	• The attacker noticed that if a deposit event was sent to the bridge smart contract the *returnVaultAssets()* function would return the "deposited" ETH to the sender. • The attacker deployed a fake contract, emitted a false event and sent it to the function, which returned real ETH to the attacker	• $\mathcal{M}_{12}$ – Whitelisting valid token contracts.
Chainswap [324], [325]	Access to critical infrastructure allowed whitelisting attacker addresses.	• On Ethereum, the attacker was able to exploit the proxy factory contract, minting tokens directly into different addresses. • There was a bug in the token cross-chain quota code. The signature node automatically increases the on-chain swap bridge quota. However, a logical code flaw allowed addresses that weren't whitelisted to automatically increase this amount.	• $\mathcal{M}_{51}$ – Smart contract auditing to identify vulnerabilities – in this case in the signature verification procedure.
pNetwork [326], [327]	The attacker emitted fake token burn events which were accepted by the source chain.	• Attackers stole BTC collateral for the pBTC-on-BSC bridge. • A series of events was created from faulty contracts, with only one being a legit peg-out request. • The attacker produced bridge back requests which were accepted. • On the original side (BTC) did not validate the requests, unlocking the funds to the attacker.	• $\mathcal{M}_{12}$ – Whitelisting valid token contracts. Events from invalid contracts should not be accepted.
Anyswap [328], [329]	The attacker exploited a bug in the signature generation algorithm.	• The attacker reconstructed the V3 router MPC account's private key due to a bug in the implementation of the random value generation algorithm. • Two transactions used the same (supposedly) random value to generate a signature which compromised the private key	• $\mathcal{M}_{76}$ – Follow standard practices, namely RFC 6979
Multichain [203]	The attacker bypassed the signature verification on the target chain.	• The function that originated the attack was not being used anywhere, it was dead code. • A fake token contract was understood by the bridge as a Multichain token. • The attacker could bypass the signature verification in the ERC-20 *permit()* function, by supplying a token contract that didn't have the permit() function implemented. This caused the fallback function to run instead of reverting. • Multichain dapp requested from all of its users a practically infinite approval sum to save gas fees, which allowed the user to withdraw practically any funds.	• $\mathcal{M}_{59}$ – Automatic tools allow identifying dead code. • $\mathcal{M}_{12}$ – Whitelisting valid token contracts. • $\mathcal{M}_{53}$ – As a user, do not grant access to all of your tokens when requested by a Dapp just to save transaction fees – coined as the baDAPProve problem
Multichain [20]	The attacker accessed internal off-chain infrastructure	• The details of this hack have not been explained yet. We refer the reader to [330] for further details. • Either a rug pull, an internal hack, or an off-chain infrastructure compromise.	We cannot propose mitigations due to not having the details of the attack
BXH [331], [332]	The private key associated with the bridge smart contract on the destination chain was compromised	• Peckshield and BXH came to the conclusion that the hacker took control over the smart contract deployed at the Binance Smart Chain after getting hold of the administrator's private key • It is suspected that this attack was an inside job using a phishing email.	• $\mathcal{M}_{70}$ – Remove a single point of failure caused by admins (or users) having excessive permissions. • $\mathcal{M}_{32}$ – Inside job attacks can be mitigated using multi-signatures or multi-party computation to decentralize responsibilities.