An Empirical Study on Cross-chain Transactions: Costs, Inconsistencies, and Activities

Kailun Yan*
Shandong University
Qingdao, China
George Mason University
Fairfax, VA, USA
kailun@mail.sdu.edu.cn

Bo Lu
George Mason University
Fairfax, VA, USA
blu5@gmu.edu

Pranav Agrawal
George Mason University
Fairfax, VA, USA
pagrawa@gmu.edu

Jiasun Li
George Mason University
Fairfax, VA, USA
jli29@gmu.edu

Wenrui Diao†
Shandong University
Qingdao, China
diaowenrui@link.cuhk.edu.hk

Xiaokuan Zhang†
George Mason University
Fairfax, VA, USA
xiaokuan.zhang.cs@gmail.com

Abstract

This paper presents the *first* large-scale measurement study on cross-chain bridges. We collected the datasets of 543,576 cross-chain transactions from four bridges, along with 1,076,972 related transactions from 11 blockchains in 2023. Using the datasets, we conducted an in-depth analysis of cross-chain transactions, focusing on their basic characteristics and costs. We also identified inconsistencies between bridge and blockchain data and performed a cluster analysis to uncover activities in cross-chain transactions. Our findings revealed 308 transactions with ledger inconsistencies that could potentially lead to asset losses. We identified four types of cross-chain activities, including 11 arbitrage bots that earned over $267k in profits within 10 months, and a liquidity pool attack that caused more than $570k in losses and led to a two-month suspension of bridge services. We tracked 24 known malicious addresses involved in cross-chain bridge activities and found 82 related transactions used for transferring illicit funds. Additionally, we uncovered six previously unreported malicious addresses.

CCS Concepts

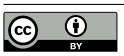
• **Security and privacy** → *Software security engineering*.

Keywords

Cross-Chain Bridge, Smart Contract, Measurement, Security

ACM Reference Format:
Kailun Yan, Bo Lu, Pranav Agrawal, Jiasun Li, Wenrui Diao, and Xiaokuan Zhang. 2025. An Empirical Study on Cross-chain Transactions: Costs, Inconsistencies, and Activities. In *ACM Asia Conference on Computer and Communications Security (ASIA CCS '25), August 25–29, 2025, Hanoi, Vietnam.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3708821.3733878

*This work was done during Kailun Yan's visit to George Mason University.
†Corresponding author.



ASIA CCS '25, Hanoi, Vietnam

ACM ISBN 979-8-4007-1410-8/25/08
https://doi.org/10.1145/3708821.3733878

1 Introduction

In recent years, there has been a surge in the popularity of various widely-used blockchain applications. These applications enable users to transfer tokens and perform computations using smart contracts, which are self-executing programs that typically function as the backend for Decentralized Finance (DeFi) Applications on blockchains (e.g., Ethereum). As of Q1 2024 [17], the Total Value Locked (TVL) in DeFi has reached $175 billion; the userbase has grown to seven million Daily Unique Active Wallets (UAW).

Although blockchain applications are widely used, their isolation prevents direct communication or data sharing across chains, hindering cross-chain token transfers. To address this, cross-chain bridges have been developed to enable interoperability between blockchains [54]. For example, Allbridge [2] enables the transfer of stablecoins between various blockchains, while Wormhole [66] utilizes a decentralized cross-chain messaging protocol to allow for the seamless transfer of assets and data between blockchains. As of September 2024, cross-chain bridges hold a TVL of around $29 billion [21]. Due to the growing popularity, low maturity, and high TVL of cross-chain bridges, bridges increasingly attract the attention of attackers. For instance, in January 2024, hackers exploited Orbit Chain, a cross-chain platform, stealing around $81 million in tokens [31]. Financial damages from attacks on cross-chain bridges amounted to $2.8 billion as of May 2024, accounting for 47% of the overall losses from all DeFi attacks [20].

Existing studies primarily focus on cross-chain technologies [12, 38, 55, 57, 59, 63, 67, 74] and security risks [25, 26, 32, 75, 76]. while blockchain measurement studies [33, 39, 61, 73, 78] provide valuable insights into on-chain transactions. In this paper, we perform the *first* large-scale measurement study on cross-chain bridges. We first collect bridge transaction data from their official APIs, then we collect transaction information from the corresponding source chain and destination chain. The data collection process yields datasets of four bridges (Allbridge, Connext, Orbit, Wormhole) with 543,576 cross-chain transactions and 11 chains with 1,076,972 on-chain transactions. The TVL of the four bridges exceeds $2.07 billion, and the TVL of the 11 chains exceeds $104.52 billion [18].

Research Questions. Using the large-scale dataset of cross-chain transactions, we aim to answer the following research questions.

- **RQ1: (Cross-chain Transaction Costs)** What are the time, gas, and transfer costs in cross-chain transactions, and how do they impact the efficiency and reliability of bridges? (§5)
- **RQ2: (Ledger Inconsistencies)** What are the inconsistencies between cross-chain bridges and the underlying blockchains, and what are the causes of those inconsistencies? (§6)
- **RQ3: (Cross-chain Activities)** What activities took place in cross-chain transactions, particularly any unusual patterns? (§7)

Findings. We have the following major findings.

- While most transactions completed within within a few minutes to three hours and only cost a few US dollars, we also observed anomalies. Connext and Wormhole had 83 and 26 transactions, respectively, with time costs exceeding one month; the maximum one exceeds 200 days. Additionally, 83 transactions had high transfer costs (over $100), with the largest cost being $4,123.
- We defined *amount inconsistency* and *unit inconsistency* that refer to a mismatch between the amounts or units in the bridge ledger and the blockchain ledgers. We found that 9,956 (1.83%) cross-chain transactions had amount inconsistencies across the four bridges; Connext had 308 transactions that were marked as successful but actually failed, posing a risk of asset loss. Moreover, three of the four bridges have unit inconsistencies.
- Our cluster analysis identified four main types of cross-chain activities: *ordinary transactions*, *large transfers*, *arbitrage bots*, and *liquidity pool (LP) attacks*. 93.15% of transactions were ordinary. However, large transfers, sometimes involving tens of thousands of dollars, caused significant market fluctuations and increased transfer costs (up to 13%). We found 11 arbitrage bots that made $267,854 in profits over ten months, and an LP attack that led to $570,000 in losses and a two-month suspension of Allbridge services. We also tracked 24 known malicious addresses involved in 82 cross-chain transactions, using bridges to transfer illicit funds. This includes an address linked to the hacker group *Pink Drainer*[1]. Our analysis also uncovered six previously unreported suspicious addresses.

Contributions. Our study makes the following contributions.

- *Large-scale cross-chain bridge transaction datasets:* We collected the *first* large-scale datasets of 543,576 cross-chain transactions from four cross-chain bridges and 1,076,972 related on-chain transactions from 11 chains, spanning the year 2023. The datasets will be open-sourced to facilitate future research[2].
- *Systematic analysis of cross-chain transactions:* We conducted an in-depth and systematic analysis of cross-chain transactions, including their basic characteristics and cost metrics. We analyzed the inconsistencies between bridge transactions and on-chain transactions, highlighting issues such as amount inconsistencies and unit inconsistencies.
- *New methods for cross-chain activity analysis:* We propose analyzing cross-chain activities through transaction graphs rather than individual transactions. We successfully identified several activities, including arbitrage and attacks. Also, we uncovered six previously unreported suspicious addresses.

[1]Pink Drainer is a notorious hacker group that has stolen over $85 million in cryptocurrency from more than 21,000 victims in year 2023 [44].
[2]The dataset is available at https://doi.org/10.5281/zenodo.15392759

Ethical Considerations. Cross-chain Data was collected via official APIs from bridge and blockchain explorers (e.g., Etherscan), strictly following their API policies. All data is publicly available, and no personally identifiable information (PII) is included.

2 Background

Blockchains are distributed ledgers that allow secure, transparent, and immutable transactions between participants in a network [69]. These systems have gained significant popularity due to their ability to support applications such as cryptocurrencies and decentralized finance (DeFi). Each blockchain operates independently, following its own consensus mechanisms and rules, which results in a lack of interoperability between different chains.

Smart contracts are self-executing code on the blockchain [5], enabling complex transactions without intermediaries and serving as the foundation for DApps [7], such as tokens [46], NFTs [19, 64], DeFi [13], swaps [4]. A contract includes some predefined events that are emitted during execution [22]. Off-chain entities, such as crypto wallets and cross-chain bridges, can listen to these events, allowing real-time tracking of on-chain activities.

Cross-chain bridges are designed to enable the transfer of assets and data between separate blockchain networks [29, 54, 56]. They unlock the ability for users to operate in a multi-chain environment, improving liquidity, and facilitating a more interconnected decentralized ecosystem. Common cross-chain technologies include:

Lock and Mint is a cross-chain mechanism where assets are locked on the source chain, and an equivalent amount of tokens is minted on the destination chain [57, 67]. This allows users to interact with assets on the destination chain, while the original assets remain locked in a smart contract on the source chain. It ensures a 1:1 peg between the locked asset and the minted token.

Burn and Release is typically the reverse of the Lock and Mint process [9, 76]. In this protocol, tokens minted on the destination chain are burned (destroyed), and the original locked assets on the source chain are released and returned to the user. This ensures that the total supply of assets across chains remains consistent.

Cross-Chain Messaging Protocols enable the transfer of data between different chains, rather than assets [8]. This is achieved using relayers, oracles, and validators to ensure message integrity, employing cryptographic verification methods such as Merkle proofs and threshold signatures to maintain cross-chain consensus and prevent fraudulent activities.

Atomic Swap is a decentralized method for exchanging assets between two different chains [28, 59]. It uses cryptographic techniques such as Hash Time-Locked Contracts (HTLC) to ensure that both parties complete the transaction simultaneously, or neither transaction occurs, eliminating the risk of fraud.

3 Overview

This section first presents the cross-chain transaction model and workflow, followed by the research questions. For convenience, we use SrcTx and DstTx for source and destination transactions. Similarly, SrcAddr and DstAddr refer to addresses, SrcChain and DstChain to blockchains, SrcTxhash and DstTxhash to transaction hashes, and SrcAmt and DstAmt to the transferred amounts.

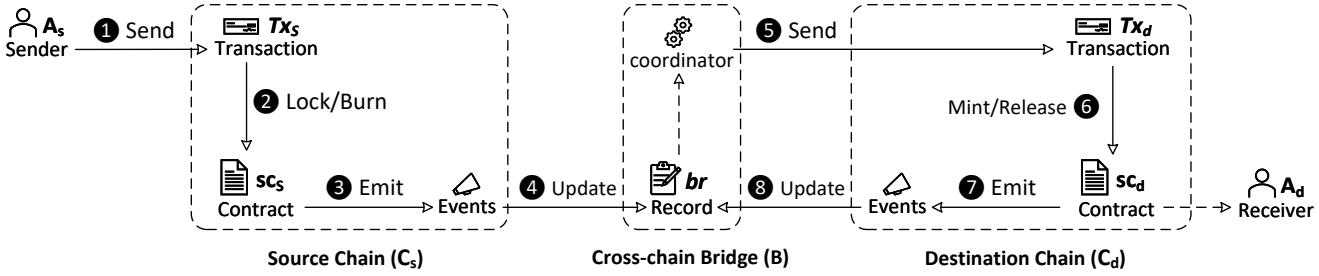


Fig. 1: Cross-chain Transaction Workflow

Table 1: Notations for Cross-Chain Transaction Analysis

Symbol	Description
$\mathbf{A}$	User address, with source $\mathbf{A}_\mathrm{s}$ and destination $\mathbf{A}_\mathrm{d}$.
$\mathbf{B}$	Bridge ledger containing records br, where $\mathbf{B} = \langle \ldots, br_i, br_{i+1}, \ldots \rangle$.
$\mathbf{C}$	Blockchain ledger containing blocks b, with source chain $\mathbf{C}_\mathrm{s}$ and destination chain $\mathbf{C}_\mathrm{d}$, where $\mathbf{C} = \langle \ldots, b_i, b_{i+1}, \ldots \rangle$.
$\mathbf{sc}$	Smart contract, with source $\mathbf{sc}_\mathrm{s}$ and destination $\mathbf{sc}_\mathrm{d}$.
b	Block containing transactions tx, with source block b_s and destination block b_d, where $b \in \mathbf{C}$ and $b = \langle \ldots, tx_i, tx_{i+1}, \ldots \rangle$.
br	Off-chain bridge record, where $br \in \mathbf{B}$ and $br = (\mathbf{A}_\mathrm{s}, \mathbf{A}_\mathrm{d}, \mathbf{C}_\mathrm{s}, \mathbf{C}_\mathrm{d}, \mathbf{sc}_\mathrm{s}, \mathbf{sc}_\mathrm{d}, \mathcal{H}(tx_\mathrm{s}), \mathcal{H}(tx_\mathrm{d}), amt_\mathrm{s}, amt_\mathrm{d})$.
tx	On-chain transaction, with source tx_s and transaction tx_d, where $tx \in b$ and $tx = (\mathbf{A}_\mathrm{s}, \mathbf{A}_\mathrm{d}, amt)$. amt is the transaction amount.
$\mathcal{H}$	Hash function, e.g., $\mathcal{H}(tx)$.
$\mathcal{T}$	Block timestamp function, e.g., $\mathcal{T}(b)$.
$\mathcal{G}$	Gas cost function, e.g., $\mathcal{G}(tx)$, with $\mathcal{G}(tx)_\mathrm{price}$ for gas price and $\mathcal{G}(tx)_\mathrm{used}$ for gas usage.

3.1 Cross-chain Model

A cross-chain transaction primarily involves three entities: the user (sender and receiver), the cross-chain bridge, and the blockchain (SrcChain and DstChain). Table 1 lists the symbols used in our model, and Fig. 1 illustrates the workflow of cross-chain transactions. Appendix A provides formalized definitions of cross-chain transactions. The main workflow is as follows:

❶ The user (sender) $\mathbf{A}_\mathrm{s}$ accesses the cross-chain bridge $\mathbf{B}$ and initiates a cross-chain transaction. The bridge prompts the user to sign the SrcTx tx_s, which is then sent to the SrcChain $\mathbf{C}_\mathrm{s}$. ❷ The SrcTx invokes a smart contract $\mathbf{sc}_\mathrm{s}$ on the SrcChain $\mathbf{C}_\mathrm{s}$, which locks or burns the user's assets amt_s. ❸ Upon completion of the asset lock or burn, the contract $\mathbf{sc}_\mathrm{s}$ emits events on the SrcChain to signal the success of the SrcTx. ❹ The bridge continuously listens for events. When it detects the relevant event, it updates its local transaction record br, noting the SrcTx details, such as the amount of assets transferred. ❺ After verifying the events and the SrcTx, the bridge's internal coordinator prepares and sends the corresponding transaction tx_d on the DstChain $\mathbf{C}_\mathrm{d}$. ❻ The DstTx tx_d calls a contract $\mathbf{sc}_\mathrm{d}$ on the DstChain $\mathbf{C}_\mathrm{d}$, which unlocks or mints the corresponding DstAmt amt_d, making it available to the receiver $\mathbf{A}_\mathrm{d}$. ❼ The contract $\mathbf{sc}_\mathrm{d}$ emits events to signal the success of the DstTx on the DstChain. ❽ The bridge monitors the DstChain for the emitted events. Upon confirming the transaction's success, it updates its local records br and notifies the user that the cross-chain transaction is complete, providing access to the assets on the DstChain.

Table 2: Key Metrics in Research Questions

	Metric	Definition
RQ1	Time cost	$\Delta t = \mathcal{T}(b_\mathrm{d}) - \mathcal{T}(b_\mathrm{s})$, where $tx_\mathrm{s} \in b_\mathrm{s}, tx_\mathrm{d} \in b_\mathrm{d}$
	Gas cost	$\Delta g = \mathcal{G}(tx_\mathrm{s})_\mathrm{used} \times \mathcal{G}(tx_\mathrm{s})_\mathrm{price} + \mathcal{G}(tx_\mathrm{d})_\mathrm{used} \times \mathcal{G}(tx_\mathrm{d})_\mathrm{price}$
	Transfer cost	$\Delta r = 1 - (tx_\mathrm{d}.amt / tx_\mathrm{s}.amt)$
RQ2	Amount inconsistency	$br.amt_\mathrm{s} \neq tx_\mathrm{s}.amt \vee br.amt_\mathrm{d} \neq tx_\mathrm{d}.amt$, where $br.\mathcal{H}(tx_\mathrm{s}) = \mathcal{H}(tx_\mathrm{s}), br.\mathcal{H}(tx_\mathrm{d}) = \mathcal{H}(tx_\mathrm{d})$
	Unit inconsistency	Type 1: $s_1 \neq s_2 \vee d_1 \neq d_2$, Type 2: $s_1 \neq d_1 \vee s_2 \neq d_2$, where $s_1 = \lfloor \log_{10}(br.amt_\mathrm{s}) \rfloor$, $s_2 = \lfloor \log_{10}(tx_\mathrm{s}.amt) \rfloor$, $d_1 = \lfloor \log_{10}(br.amt_\mathrm{d}) \rfloor$, $d_2 = \lfloor \log_{10}(tx_\mathrm{d}.amt) \rfloor$, $br.\mathcal{H}(tx_\mathrm{s}) = \mathcal{H}(tx_\mathrm{s})$, $br.\mathcal{H}(tx_\mathrm{d}) = \mathcal{H}(tx_\mathrm{d})$
RQ3	Arbitrage	$br.amt_\mathrm{d} - br.amt_\mathrm{s} > 0$

3.2 Research Questions

From the perspective of cross-chain transactions, this paper proposed three key research questions, which target several critical aspects of cross-chain transactions. Table 2 summarizes the key metrics related to our research questions, and we will explain these metrics in detail in later sections.

RQ1: Cross-chain Transaction Costs (§5). We aim to provide a comprehensive understanding of the costs associated with cross-chain transactions. By collecting real on-chain data, we measure the time cost, gas cost, and transfer cost. These metrics are crucial for users and developers to evaluate the performance and feasibility of cross-chain transactions.

RQ2: Ledger Inconsistencies (§6). Ledger inconsistencies in cross-chain bridges create development challenges, undermine user trust, and introduce security risks. Discrepancies between off-chain and on-chain data can cause confusion, miscalculations, and potential financial losses while increasing development complexity. By analyzing these issues, we aim to enhance the transparency, security, and reliability of cross-chain bridges.

RQ3: Cross-chain Activities (§7). Cross-chain transactions differ significantly from normal transactions, potentially creating opportunities for specific use cases or introducing risks. By examining transaction activity within cross-chain operations, we aim to uncover activities and summarize unusual patterns, such as arbitrage, which could inform the design of more secure and efficient systems. Additionally, we investigate the activities of known malicious addresses in cross-chain transactions, providing insights into security threats within the blockchain ecosystem.

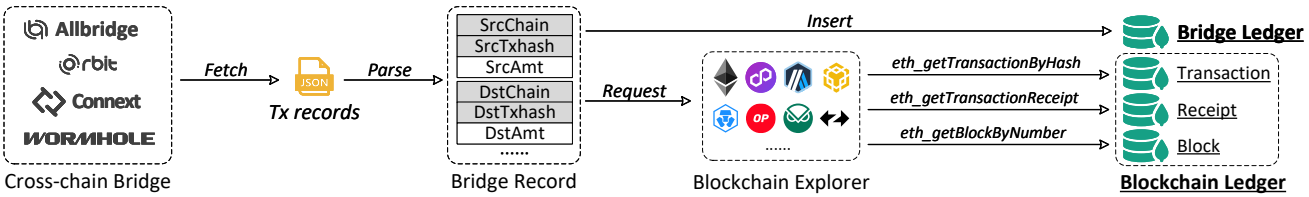


Fig. 2: Data Collection Process

4 Data Collection

This section first introduces the selection of cross-chain bridges and blockchains, then describes the data collection process, and finally provides an overview of our dataset.

4.1 Selection of Bridges and Chains

We selected four cross-chain bridges and 11 blockchains to comprehensively evaluate cross-chain transactions.

Cross-chain bridges. We investigated 30 widely used cross-chain bridges [76], covering both blockchain-to-Layer-2 (L1-L2) bridges and cross-blockchain (L1-L1) bridges. Appendix §B.1 details the bridges and their APIs. Our data collection relied on two APIs:

- **Tx Info API**, which retrieves details of a specific cross-chain transaction using a transaction hash.
- **Tx Index API**, which provides an index of transactions over a given time period (e.g., a year).

As shown in Table 13, 11 bridges provide the Tx Info API. However, without the Tx Index API, they do not offer a way to obtain cross-chain transaction hashes, making it impossible to retrieve all transactions within a specific timeframe. Only four bridges provide both APIs, making them suitable for our analysis: Allbridge [2], Connext [15], Orbit [48], and Wormhole [66]. These bridges collectively hold over $2.07 billion in Total Value Locked (TVL) and employ diverse cross-chain mechanisms. In terms of architecture: Allbridge and Wormhole primarily operate on-chain, with off-chain components for relayers or coordination. Orbit and Connext function off-chain, with final transaction states committed to the blockchain. For cross-chain transfers: Allbridge uses a Lock and Mint model with liquidity pools and also supports Wormhole's underlying protocol [1]. Connext employs state channels for fast, low-cost transactions [14]. Orbit utilizes a cross-chain messaging protocol for asset and data transfers [47]. Wormhole relies on messaging protocols to archive assets transfer [65]. Regarding asset types, Allbridge supports only stablecoins (ERC20), while Connext supports various ERC20 tokens and NFTs. Orbit and Wormhole handle a broader range of ERC20 tokens. Since ERC20 tokens dominate cross-chain transactions, our study focuses on ERC20 transfers.

Chains. We collected on-chain data to comprehensively evaluate cross-chain transactions. Public blockchain explorers, like Etherscan, offer APIs for accessing on-chain data. The *blockscan.com* lists 22 blockchain explorers developed by the Ethereum team. We matched these 22 blockchains with those supported by the four bridges and identified 11 compatible blockchains as shown in Table 4. As of September 2024, the TVL of these 11 blockchains exceeded $63 billion [18].

4.2 Data Collection Process

We first collected the public APIs of the selected cross-chain bridges and blockchains. Then, we designed a tool to request these APIs to collect data. Fig. 2 describes our data collection process, which consists of the following steps:

Step 1: Fetching and parsing cross-chain transactions: Our tool uses the APIs provided by cross-chain platforms to fetch cross-chain records from four bridges. These records contain basic cross-chain information, such as SrcChain, DstChain, SrcTxhash, DstTxhash, SrcAmt, and DstAmt. In Table 1, *br* presents the main fields.

To handle the different formats of cross-chain records, we added a configuration file to map key fields for each bridge. Our tool first parses key fields from the transaction records, such as amounts (SrcAmt, DstAmt), chains (SrcChain, DstChain), and transaction hashes (SrcTxhash, DstTxhash), then stores the data in the database, referred to as the *bridge ledger*. This creates uniform ledgers for the four bridges, simplifying analysis.

Step 2: Fetching source and destination transactions from chains: Using the bridge ledger, our tool fetched on-chain data for cross-chain transactions on both SrcChain and DstChain. It selects the appropriate blockchain explorers based on SrcChain and DstChain, then fetches on-chain data using SrcTxhash and DstTxhash. The data is stored in the *blockchain ledger*. We utilize APIs provided by blockchain explorers [23] to request *transactions*, *receipts*, and *blocks*, using the following three APIs:

- *eth_getTransactionByHash* returns details for a given transaction hash, including the *sender's address*, *recipient's address*, *transferred amount*, *block number*, and *gas price*. This data is key for analyzing transaction costs.
- *eth_getTransactionReceipt* returns the transaction receipt for a given transaction hash. The receipt includes *gas used*, *logs*, etc. The transaction-emitted events are recorded in *logs*, which are essential for tracking transaction outcomes.
- *eth_getBlockByNumber* returns detailed information about a block given its block number, such as the *blockhash* and *timestamp*. We use the *timestamp* as the time of the transaction.

4.3 Datasets

We collected 543,576 cross-chain transactions (records) from four cross-chain bridges-Allbridge, Connext, Orbit, and Wormhole —spanning from January 1, 2023 to December 31, 2023 (one year). Additionally, we gathered 1,076,972 related on-chain records from 11 blockchains. The data collection followed the guidelines and rate limits of the bridge websites and blockchain explorers. Due to API rate limits, the entire collection process took two months. The dataset consists of two parts:

Table 3: Number of Cross-chain Transactions on Four Bridges

Allbridge	Connext	Orbit	Wormhole	Total
54,587	453,165	2,763	33,061	543,576

Table 4: Number of Transactions on 11 Chains

	Allbridge	Connext	Orbit	Wormhole	Total
Polygon	44,332	260,712	2,112	14,380	321,536
Arbitrum	11,746	207,662	N/A	6,996	226,404
Binance	38,552	150,621	1,070	N/A	190,243
Optimism	N/A	132,530	N/A	5,681	138,211
Gnosis	N/A	122,435	6	N/A	122,441
Ethereum	4,642	31,970	1,031	6,564	44,207
Celo	N/A	N/A	58	12,952	13,010
Fantom	N/A	N/A	2	11,082	11,084
Base	N/A	N/A	N/A	8,375	8,375
Wemix	N/A	N/A	1,247	N/A	1,247
Linea	N/A	214	N/A	N/A	214
Total	99,272	906,144	5,526	66,030	1,076,972

N/A: the bridge does not support the chain or the transactioncount is zero.

- **Bridge Ledger.** This dataset contains 543,576 cross-chain records from four bridges, reflecting the local bridge data. Each entry includes basic cross-chain transaction details like SrcTx, DstTx, SrcAmt, DstAmt, SrcAddr and DstAddr. However, Orbit and Wormhole only provide one amount, so the same value is used for both transactions.
- **Blockchain Ledger.** This dataset includes 1,076,972 on-chain records from 11 chains, reflecting the actual on-chain data of the cross-chain transactions. The on-chain data includes transactions, receipts, and blocks.

Table 3 presents cross-chain transactions across four bridges and 11 chains in 2023, while Table 4 includes only those with available on-chain data. We use these datasets for analysis in §5.3 and §6. To extract the actual transferred amounts, we rely on TransferEvents from on-chain transactions, as detailed in Appendix §B.2. Since some records share the same SrcTx and DstTx, the on-chain transaction count in Table 4 is not exactly twice that in Table 3.

Additionally, Allbridge has four chains that are not listed in Table 4 because blockscan.com does not provide APIs for these chains. Nonetheless, we collect their bridge records, totaling 146,210, and use them for cross-chain activity analysis in §7.

5 Cross-chain Transaction Cost

This section presents the basic metrics of cross-chain bridges, including time cost, gas cost and transfer cost.

5.1 Time Cost

The *time cost* of a cross-chain transaction refers to the period between the sent of SrcTx and the completed of DstTx. This metric is crucial for evaluating the performance of cross-chain bridges as it directly impacts user experience. However, on-chain transactions do not have timestamps, and the timestamps in cross-chain records are incomplete. For example, Allbridge only records the creation time, while Wormhole omits the time of SrcTx. As shown in Table 2, we use timestamps of the blocks of SrcTx and DstTx to calculate the time cost Δt, which is an approximation.

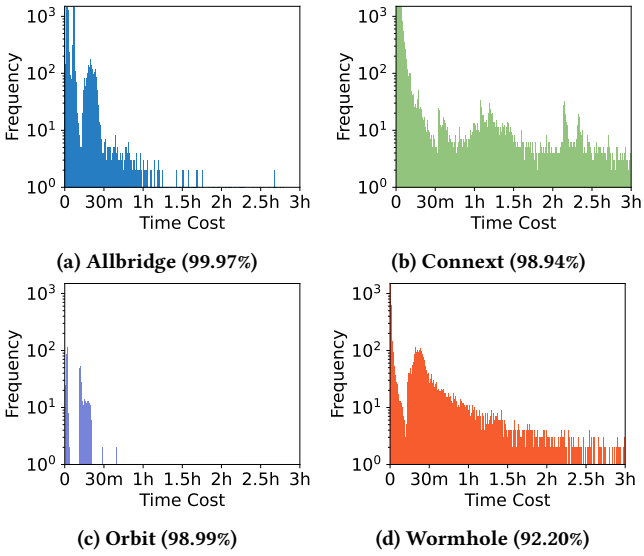
$$\Delta t = \mathcal{T}(b_{\mathrm{d}}) - \mathcal{T}(b_{\mathrm{s}}) \tag{1}$$

where $tx_s \in b_{\mathrm{s}}$, $tx_d \in b_{\mathrm{d}}$.

Table 5: Time cost of Cross-Chain Transactions (99%)

	Min	Max	Mean	Median	Std
Allbridge	0s	37m 35s	6m 19s	4m 40s	6m 43s
Connext	20s	3h 15m	3m 47s	1m 34s	13m 37s
Orbit	46s	3h 12m	9m 58s	11m 38s	13m 42s
Wormhole	10s	2d 21h	1h 25m	20m 5s	5h 17m

d: day, h: hour, m: minute, s: second



(a) Allbridge (99.97%) (b) Connext (98.94%)

(c) Orbit (98.99%) (d) Wormhole (92.20%)

Fig. 3: Distribution of Time Cost (≤ 3 Hours)

Result Analysis. Our results can be considered a lower bound for actual transactions. Appendix C presents the cumulative distribution function (CDF) of time cost, with Fig. 10 showing most transactions complete within 30 minutes. To ensure a more accurate representation of typical transaction times, we excluded the slowest 1% of transactions across all four bridges, as Wormhole allows user-submitted DstTxs, which can introduce delays. Table 5 summarizes the time cost for the remaining 99% of transactions. Allbridge is stable (mean and std 6 min). The median of Connext (1m 34s) is far below its mean (3m 47s), indicating quick completions. The median of Orbit (11m 34s) exceeds its mean (9m 58s), suggesting frequent delays. Wormhole shows high variability, with a maximum time of nearly three days (2d 21h).

Fig. 3 illustrates the time cost distribution. About 99% of transactions on Allbridge, Connext, and Orbit complete within three hours, while 7.80% of Wormhole transactions take longer. We observe distinct peaks in Fig. 3, primarily influenced by transaction routes and network conditions. Wormhole has two peaks because it supports paying higher fees for faster cross-chain routes. Allbridge shows three peaks as it uses both its own protocol and Wormhole's infrastructure [1]. Orbit's peaks arise from varying time costs across chains—15 minutes for ETH and Celo, 4 minutes for others. Low transaction volume further amplifies this effect, whereas other bridges' higher volume smooths variations.

Outlier Analysis. We identified some unusual time costs in our results. As shown in Table 5, Allbridge recorded a minimum time

Table 6: Gas Cost (Gwei) for Cross-Chain Transactions

	Min	Max	Mean	Median	Std
Allbridge	2.03×10^5	6.17×10^9	3.66×10^7	2.36×10^7	6.40×10^7
Connext	3.09×10^4	3.39×10^9	4.08×10^7	3.28×10^7	5.71×10^7
Orbit	6.44×10^5	2.99×10^8	3.73×10^7	3.12×10^7	3.03×10^7
Wormhole	7.19×10^{-1}	7.92×10^9	2.85×10^7	1.20×10^7	1.14×10^8

1 Ether = 10^9 Gwei = 10^{12} Mwei = 10^{15} Kwei = 10^{18} Wei.

cost of zero seconds. This cross-chain transaction involved the same SrcTx and DstTx. In §7, we identified this as an attack and provide a detailed analysis. Additionally, Connext and Wormhole had 83 and 26 transactions, respectively, that took over a month to complete. After manual verification, we confirmed these were valid cross-chain transactions, with delays primarily caused by late submission of the DstTx. In the case of Wormhole, users can manually submit DstTx, leading to delays if they do not act promptly. Moreover, OKX notes that the bridge may not always immediately receive the status of the SrcTx, further delaying cross-chain transfers [45].

Finding 1. Most cross-chain transactions take just a few minutes, with the majority completed within three hours. The time cost varies significantly across different bridges, largely due to the protocols they use. Additionally, an attack transaction have a very short time cost.

5.2 Gas Cost

Gas cost, or gas fee, is the transaction fee users pay for executing transactions on a blockchain network. An on-chain transaction's gas cost is determined by the *gas used* and the *gas price*. The gas used is the amount of gas consumed to execute a transaction, while the gas price is the amount users are willing to pay per unit of gas. In Table 2, the gas cost Δg of a cross-chain transaction is the sum of the gas costs of SrcTx and DstTx.

$$\Delta g = \mathcal{G}(tx_s)_{\text{used}} \times \mathcal{G}(tx_s)_{\text{price}} + \mathcal{G}(tx_d)_{\text{used}} \times \mathcal{G}(tx_d)_{\text{price}} \quad (2)$$

Gas used reflects the efficiency of cross-chain contracts. The median gas used for all four bridges is around 10^5 Gwei. Compared to typical *transfer* transactions with 2.1×10^4 gas used [41], cross-chain transactions consume more due to the complexity of contracts and the involvement of two on-chain transactions. However, the median gas used is similar to *swap* transactions on Uniswap, which use 1.7×10^5 gas [58]. Some transactions, especially on Wormhole, use significantly more gas, with a maximum of 2.52×10^8 Gwei, indicating potentially higher costs compared to other bridges.

Gas price reflects blockchain activity and transaction costs, with higher prices indicating busier networks and increased costs. To compare gas prices across chains, we converted them to USD based on the respective cryptocurrency's value. Ethereum has significantly higher gas prices, with a median of 4.97×10^{-5} USD, while other chains range from 1.20×10^{-12} to 1.33×10^{-7} USD.

Gas cost is a crucial factor. Each cross-chain transaction consists of a SrcTx and a DstTx, with each blockchain charging a gas fee for its portion. Table 6 shows the sum of SrcTx and DstTx gas costs, with median costs for all four bridges fall within the range of 10^7 Gwei. Fig. 4 shows the distribution of gas costs. The histograms indicate that Allbridge, Connext, and Orbit have stable gas costs, with most

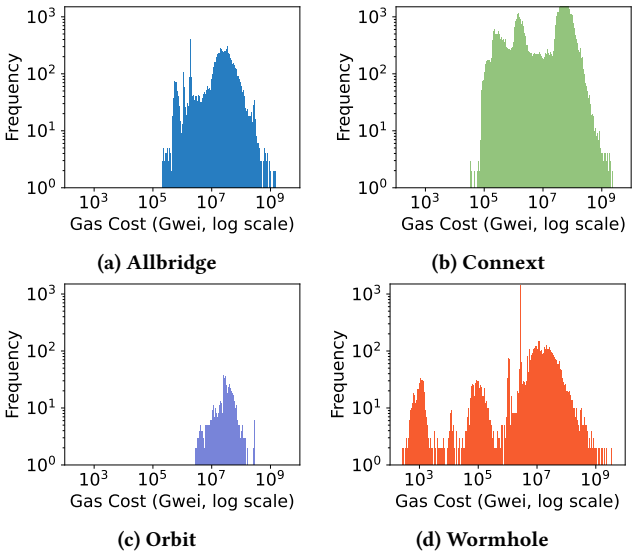


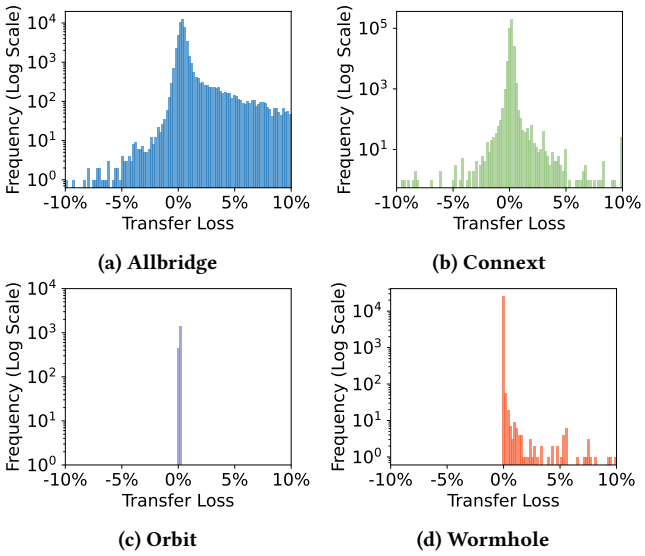
Fig. 4: Distribution of Gas Cost Across Four Bridges

transactions ranging from 10^5 to 10^9 Gwei. Notably, Wormhole shows some significantly lower gas costs, as it supports custom relayers that offload on-chain computations to off-chain processing, reducing gas usage [65]. However, some transactions have a gas cost exceeding 1 Ether (10^9 Gwei). Additionally, the maximum values in Fig. 4 show that the highest gas costs for Allbridge, Connext, and Wormhole are more than 100 times higher than the mean and median values. Upon analyzing these transactions, we identified two main reasons for this: 1) the gas prices on certain chains are inherently high leading to higher gas costs; For example, we found when cross-chain transactions involve Ethereum or Polygon, the gas costs are significantly higher than others. and 2) some users set a higher gas price to speed up transactions. In the Polygon chain, we found six transactions with gas costs significantly higher (100×) than the others. These six transactions were identified as arbitrage transactions in §7. Specifically, arbitrager set a significantly higher gas price (10×) than typical gas prices on the Polygon chain to ensure these transactions were processed as quickly as possible to achieve the arbitrage goal.

Finding 2. The median gas costs are around 10^7 Gwei (a few USD), but some transactions incur high costs (e.g., >1 Ether) for time-sensitive operations like arbitrage. Offloading non-critical on-chain computations to off-chain processing can significantly reduce gas usage.

5.3 Transfer Cost

In cross-chain transfer, asset loss due to bridge fees, exchange rate slippage, and other factors is called the *transfer cost*. Orbit and Connext support multiple tokens with fluctuating USD prices, but neither provides the USD value of the transferred amounts. As a result, we cannot directly compare the transfer costs. To standardize the measurement, we calculate the transfer cost Δr using the ratio



(a) Allbridge (b) Connext

(c) Orbit (d) Wormhole

Fig. 5: Distribution of Tranfer Cost (-10% ~ 10%)

of DstAmt to SrcAmt:

$$\Delta r = 1 - (tx_{\mathrm{d}}.amt / tx_{\mathrm{s}}.amt) \tag{3}$$

$\Delta r = 0$ indicates no asset loss, while $\Delta r = 1$ represents total asset loss. A negative Δr indicates asset gains, possibly due to exchange rate differences [2] or arbitrage, which we will discuss in §7.

Allbridge and Connext provide both SrcAmt and DstAmt, allowing direct calculation of transfer costs from ledger data. In contrast, Orbit and Wormhole expose only one amount, so we extracted SrcAmt and DstAmt from their TransferEvents. This yielded 1,861 pairs from 2,763 Orbit transactions (67.4%) and 25,189 pairs from 33,061 Wormhole transactions (76.2%). The remaining transactions lacked TransferEvents and could not be processed.

Result Analysis. As shown in Fig. 5, the distribution of transfer losses varies across bridges. The y-axis is in log scale, therefore, most transfer losses cluster slightly above zero. However, Allbridge and Connext show a wider spread due to liquidity disparities across chains, creating arbitrage opportunities. Allbridge also has many low-value transactions, leading to more frequent high transfer losses. Wormhole has the highest at 0.81%, while Connext and Orbit have lower costs at 0.15% and 0.10%, respectively. Allbridge's median is 0.43%. Most Wormhole transfers show zero cost since users can pay fees from their destination wallets. Overall, cross-chain transfer costs are low. In Allbridge, 50% of transactions cost under $0.34, and 90% under $3.24. However, 83 transactions exceeded $100, with the highest at $4,123.26 (13% loss). These high costs, driven by large transfers, lead to liquidity shortages on the destination chain, as shown in §7.

Finding 3. The median transfer cost are within 1% across the four bridges, indicating that most transactions benefit from the high efficiency of current cross-chain transfers. However, liquidity fluctuations can lead to higher transfer costs for users.

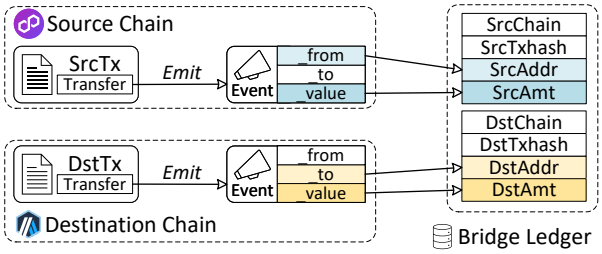


Fig. 6: Inconsistency Between Bridge and Chain Ledgers

6 Ledger Inconsistencies

Cross-chain transactions involve three ledgers: the bridge ledger, the SrcChain ledger, and the DstChain ledger. Inconsistencies between these ledgers could lead to security risks and asset loss. This section addresses two types of inconsistencies: *Amount Inconsistency* and *Unit Inconsistency*. These inconsistencies create challenges in protocol functioning and increase the likelihood of confusion and attack surfaces. For example, in Connext, we have observed 308 amount inconsistencies may have led to asset loss. Additionally, Wormhole supports custom relayers, and unit inconsistencies complicate relayer design, as developers must address these discrepancies manually, increasing security risks.

Amount Inconsistency occurs the amounts recorded in the bridge ledger differ from those in the blockchain ledgers. As outlined in §3.1, the bridge ledger tracks SrcAmt in step ❹ and DstAmt in step ❼. The SrcAmt should match the amount in the SrcChain ledger, while DstAmt should match the amount in the DstChain ledger. Fig. 6 illustrates the relationship between the amounts in the bridge and blockchain ledgers: the *value* (SrcValue) in the SrcTx's TransferEvent should match the SrcAmt in the bridge ledger, and the *value* (DstValue) in the DstTx's TransferEvent should match the DstAmt. Amount inconsistency can be represented as:

$$br.amt_{\mathrm{s}} \neq tx_{\mathrm{s}}.amt \ \vee \ br.amt_{\mathrm{d}} \neq tx_{\mathrm{d}}.amt \tag{4}$$

where $br.\mathcal{H}(tx_{\mathrm{s}}) = \mathcal{H}(tx_{\mathrm{s}})$, $br.\mathcal{H}(tx_{\mathrm{d}}) = \mathcal{H}(tx_{\mathrm{d}})$.

Unit Inconsistency refers to the inconsistency between the units of the amounts in the bridge ledger and blockchain ledgers. Through manual analysis of the bridge ledgers and their corresponding SrcValue and DstValue in TransferEvents, we found that the units of these amounts can be inconsistent. For example, in Allbridge, a transaction transferring USDC from Polygon to Binance records SrcAmt and DstAmt in the bridge ledger as 267.909851 and 267.5690435, respectively. However, the SrcValue on Polygon is 267909851, while the DstValue on Binance is 267569043500000000000. This reveals two types of unit inconsistencies:

- *Type 1: <Bridge, Chain>* - inconsistency between the bridge ledger and blockchain ledgers.
- *Type 2: <SrcChain, DstChain>* — inconsistency between the source and destination chains.

Unit inconsistency can be represented as:

$$\text{Type 1: } s_1 \neq s_2 \vee d_1 \neq d_2, \quad \text{Type 2: } s_1 \neq d_1 \vee s_2 \neq d_2 \tag{5}$$

$s_1 = \lfloor \log_{10}(br.amt_{\mathrm{s}}) \rfloor$, $s_2 = \lfloor \log_{10}(tx_{\mathrm{s}}.amt) \rfloor$, $d_1 = \lfloor \log_{10}(br.amt_{\mathrm{d}}) \rfloor$, $d_2 = \lfloor \log_{10}(tx_{\mathrm{d}}.amt) \rfloor$, $br.\mathcal{H}(tx_{\mathrm{s}}) = \mathcal{H}(tx_{\mathrm{s}})$, $br.\mathcal{H}(tx_{\mathrm{d}}) = \mathcal{H}(tx_{\mathrm{d}})$.

Table 7: Amount Inconsistency

	AllBridge	Connext	Orbit	Wormhole
SrcAmt ≠ SrcValue	7,321 (13.41%)	328 (0.072%)	1,638 (59.28%)	N/A
DstAmt ≠ DstValue	0 (0.0%)	20 (0.004%)	N/A	669 (2.02%)

N/A: ledger does not provide SrcAmt or DstAmt.

Table 8: Unit Inconsistency

	AllBridge	Connext	Orbit	Wormhole
Type 1: <Bridge, Chain>	100%	0.0%	0.0%	86.05%
Type 2: <SrcChain, DstChain>	50.65%	10.78%	0.0%	0.0%

Detection. For each cross-chain transaction, we 1) extract SrcValue and DstValue from the TransferEvents of SrcTx and DstTx and 2) compare them with SrcAmt and DstAmt in the bridge ledger. If they do not match, it indicates an *amount inconsistency*. To prevent false positives caused by unit inconsistencies, we first standardize the magnitude of the two amounts before comparison, as shown in algorithm 1. If they still differ, we confirm an *amount inconsistency*. We use the CountMagnitude() method to determine the magnitude of amounts. If the magnitudes of SrcAmt and SrcValue differ, it indicates a *unit inconsistency*. Since Orbit and Wormhole provide only one amount, we compare it with both SrcValue and DstValue.

Result of amount inconsistency detection. Table 7 shows the amount inconsistencies. Allbridge and Orbit exhibit 13.41% and 59.28% SrcAmt inconsistencies, respectively, caused by the Swap operation. Both record only the amount after the swap, leading to discrepancies. Specifically, the SrcTx swaps one token (tokenA) for another (tokenB), and the DstTx receives tokenB. However, only the amounts of tokenB are recorded in the ledgers, while the user transfers tokenA in the SrcTx, resulting in SrcAmt inconsistencies.

Connext has 0.072% SrcAmt and 0.004% DstAmt inconsistencies. These are due to two main causes: 1) *Failed DstTx*: approximately 308 transactions failed due to issues like *out of gas* or *execution reverted*. The SrcTx executed successfully with non-zero SrcValue, but Connext recorded SrcAmt and DstAmt as zero, causing inconsistencies. Additionally, these transactions are labeled as *SUCCESS* on the Connext website, indicating that Connext may not properly handle transaction exceptions, which could result in user losses. 2) *NFT transfers*: about 20 inconsistencies were caused by NFT (ERC721) transfers being mistaken for ERC20. While both use the same event signature, ERC20 uses the third parameter for value, whereas ERC721 uses it for tokenID. The reason is Connext did not differentiate them.

Wormhole only records one amount and supports two types of transactions: 1) *Approve transfer with destination wallet* requires users to manually receive and pay fees on the destination wallet, resulting in consistency between SrcAmt and DstAmt. 2) *Receive tokens automatically* deducts bridge fees during the cross-chain transfer, resulting in transfer loss and leading to DstAmt inconsistency. We found 2.02% of such transactions, as shown in Table 7.

Finding 4. We identified 9,956 (1.83%) transactions with inconsistencies, and 308 failed DstTx in Connext went undetected, potentially causing asset loss. Inconsistencies arise from bridges overlooking edge cases, such as on-chain exceptions. These inconsistencies introduce security risks and poor experience.

Result of unit inconsistency detection. Table 8 shows the unit inconsistencies between ledgers. Allbridge has *Type 1: <Bridge, Chain>* inconsistencies because it converts transferred amounts (stablecoins) into dollar units, making all amounts in the bridge ledger inconsistent with on-chain transactions. Wormhole has 86.05% of transactions with *Type 1* unit inconsistencies, with chaotic unit usage—some amounts being smaller and others larger than on-chain values. Wormhole supports custom relayers, and unit inconsistencies require developers to manually address these discrepancies, adding extra workload and increasing security risks. Additionally, Allbridge and Connext show 50.65% and 10.78% of transactions with *Type 2: <SrcChain, DstChain>* unit inconsistencies, indicating non-uniformity in the units used in their cross-chain contracts. In contrast, Orbit has no unit inconsistencies, and Wormhole maintain consistent units in cross-chain contracts.

Finding 5. Three of the four bridges have unit inconsistencies, indicating that cross-chain bridges do not properly standardize the cross-chain bridges and on-chain contracts. These inconsistencies complicate cross-chain transfers, requiring developers to handle them manually.

7 Cross-chain Activities

This section focuses on activities within cross-chain transactions, particularly unusual patterns. Given the unique characteristics of cross-chain transactions, we propose extracting features from transaction graphs rather than individual transactions and then performing clustering analysis. Our method uncovers four main types of activities: *ordinary transactions*, *large transfers*, *arbitrage bots*, and *liquidity pool (LP) attacks*. Additionally, we trace activities related to malicious addresses in cross-chain transactions and find that these addresses utilize cross-chain bridges to transfer assets. We also identify six previously unreported suspicious malicious addresses.

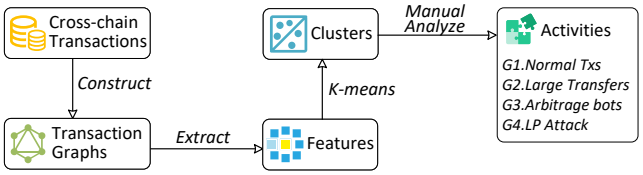


Fig. 7: Cross-chain Activity Clustering Workflow

7.1 Overview

RQ3 aims to identify and characterize activities in cross-chain transactions, particularly unusual ones. As illustrated in Fig. 7, we construct transaction graphs from collected transactions, extract relevant features, and apply clustering to group similar transaction patterns. Finally, we manually analyze the clustering results to categorize activity types.

Dataset. Due to missing on-chain data for some blockchains, transaction graphs would be incomplete. Therefore, we rely solely on ledger data for a comprehensive year-long analysis of cross-chain activity. Among the four bridge ledgers, Allbridge is the only one providing complete SrcAmt and DstAmt data, as it exclusively

supports stablecoins with a USD-equivalent value, enabling standardized analysis. In contrast, Connext supports various tokens and NFTs, complicating value measurement, while Orbit and Wormhole lack SrcAmt or DstAmt data. The Allbridge ledger contains 146,210 transactions across eight blockchains, four of which are not analyzed in §5 and §6 due to the absence of on-chain data.

7.2 Methodology

Our methodology combines graph-based feature extraction with clustering method to identify cross-chain activities.

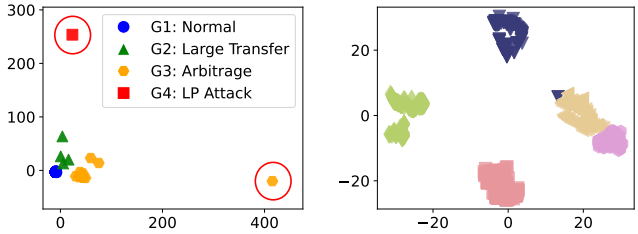
Transaction Graph. Cross-chain transactions are significantly different from other conventional transactions like bank transfers or native blockchain transaction. In conventional transactions, the sender and the receiver are usually different entities. In cross-chain transactions, the sender and receiver are often the same user or organization [36], as the primary purpose is to transfer assets across blockchains. To accommodate this, we use cross-chain transactions to construct directed multi-edge graphs, where nodes represent the SrcAddr and DstAddr and edges carry specific transaction information, such as amount. Then, we consider each graph to encapsulate the activities of a single user or organization. From 146,210 transactions, we ultimately constructed 59,465 distinct transaction graphs.

Feature Selection. To capture the activity patterns of each user or organization, we extracted features from the transaction graphs. Specifically, we focus on the four aspects: *transaction frequency*, *amounts*, *transfer cost* and *graph attribute*. Appendix §E.1 details all considered features and their rationale. We select 11 key features reflect various aspects of user activity and behavior within the transaction graph. Specifically, *TxCount* reflects the user's overall activity level, while *WkFreqMax* highlights periods of peak activity. *SrcAmtMax* and *DstAmtMax* capture the largest individual transactions, highlighting extreme cases. These are particularly useful for detecting anomalies or unusually activities. In the *transfer cost*, we use three features *ProfitNum*, *ProfitSum*, and *LossSum* to represent it. These features reflect the financial outcomes of the transactions. We also select four structural features from the graph. *DegAvg* reflects the transaction frequency of the overall node. *DegGini* measures the inequality in transaction distribution among addresses, identifying whether a few nodes dominate the activity. *AvgSPL* reflects the breadth and efficiency of the transaction network. *Density* indicates the overall interconnectedness of the graph, showing how tightly knit the network is.

K-means Clustering. We opted for the K-means clustering algorithm due to its efficiency and suitability for medium-sized datasets. K-means partitions data into k clusters by minimizing the within-cluster sum of squares (inertia), effectively grouping similar data points based on feature proximity. In addition, K-means can also be used to detect outliers when most data points form clusters and outliers are far away from the clusters. This is consistent with our goal of discovering unusual activities. Other methods such as DBSCAN and isolation forest were considered but deemed less appropriate for our goal (explained in Appendix §E.2).

Choice of k. Choosing the appropriate number of clusters k is crucial for effective K-means clustering. To determine the optimal k, we tested values ranging from 2 to 30 using the Elbow Method. We observed a noticeable inflection point between 7 and 10 clusters and selected $k = 10$ to capture as many distinct activity patterns as possible (details in Appendix §E.3).

Evaluation. Silhouette Scores assess how well a data point fits within its assigned cluster relative to others, with values ranging from -1 to 1. A score near 1 indicates strong cohesion within its cluster, around 0 suggests the point is near a cluster boundary, and close to -1 implies potential misassignment. Our clustering achieved an average Silhouette Score of 0.7094, indicating well-defined clusters and proper grouping of data points, which reflects the effectiveness of our method.



(a) 2D PCA of Four Groups (b) t-SNE of Group 1

Fig. 8: Cluster Results Visualization (100 samples)

7.3 Results Analysis

We identified ten clusters representing different cross-chain activities and manually analyzed their transactions to infer the activities. Finally, we categorized these 10 clusters into four groups: *G1: Normal Transactions*, *G2: Large Transfers*, *G3: Arbitrage Bots*, and *G4: Liquidity Pool (LP) Attack*. This section delves into each group's characteristics, focusing particularly on anomalies and implications.

Table 9 summarizes the number of transaction graphs (each representing the activity of a single user or organization) and the total number of transactions within each group and cluster. While the majority of cross-chain activities are normal asset transfers (G1), we identified four instances of large fund transfers (G2) involving 138 transactions. Additionally, we found 11 cases related to arbitrage bots (G3), comprising 9,363 transactions. Notably, we discovered a liquidity pool attack (G4) targeting the liquidity pool of Allbridge, resulting in a profit exceeding $570,000 and causing Allbridge to suspend its services for nearly two months.

Fig. 8a shows the scatter plot of clustering results after reducing the 11 features to two dimensions using Principal Component Analysis (PCA) on a sample of 100 transactions. The long-tail feature distribution and two outliers compress the axes, but the distinction between groups remains clear, and there is good differentiation within each group as well. For example, Fig. 8b provides a t-SNE visualization of five clusters within G1. Although there is one outlier, overall the clusters within the group are well distinguished. Note that setting k to a smaller value (e.g., $k = 5$) does not guarantee better clustering. Transactions in different clusters within the same group still show significant differences, as seen in Table 10. Next, we will analyze the activity of four groups in detail.

G1: Normal Transactions. In cross-chain transactions, 93.15% of activities are normal transactions. They are divided into five cluster labels (0, 3, 5, 7, 9). As shown in Table 10, these transactions

Table 9: Graph and Transaction Statistics by Cluster Group

Group	G1					G2	G3			G4
Cluster ID	0	3	5	7	9	4	1	8	6	2
Graph Count	29,297	23,258	634	6,206	58	4	1	3	7	1
Address Count	58,675	23,258	1,611	17,273	154	8	6	28	15	2
Transaction Count	52,153	38,683	3,175	2,106	506	138	4,660	2,597	2,106	1
Total Graph Count	59,453					4	11			1
Total Address Count	100,971					8	49			2
Total Transaction Count	96,623					138	9,363			1

Table 10: Cluster Label vs Feature Normalized Heatmap

Group	G1					G2	G3			G4
Cluster	0	3	5	7	9	4	1	8	6	2
TxNum*	0.1	0.1	0.2	0.2	0.3	0.4	1.0	0.8	0.7	0.1
WkFreqMax*	0.1	0.1	0.2	0.2	0.3	0.5	1.0	0.7	0.7	0.1
SrcAmtMax*	0.5	0.4	0.8	0.5	0.8	0.9	0.8	0.8	0.7	0.8
DstAmtMax*	0.5	0.4	0.8	0.5	0.8	0.9	0.8	0.8	0.7	1.0
ProfitNum*	0.0	0.0	0.1	0.1	0.0	0.1	1.0	0.8	0.5	0.1
ProfitSum*	0.0	0.0	0.4	0.2	0.6	0.5	0.9	0.8	0.6	1.0
LossSum*	0.2	0.1	0.4	0.2	0.6	0.8	0.6	0.7	0.4	0.0
DegAvg	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.1	0.2	0.0
DegGini	0.0	0.0	0.1	0.4	0.1	0.0	1.0	1.0	0.3	0.0
AvgSPL	0.5	0.0	0.6	0.6	0.6	0.5	0.8	1.0	0.5	0.5
Density	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.1	0.8	0.0

* indicates log-transformed for better data comparison.

are categorized into five types mainly because of differences in their transaction counts (*TxNum*) and the magnitude of transferred amounts (*SrcAmtMax*, *DstAmtMax*).

In this group, clusters 0, 3, and 7 have 90% of transactions not exceeding $1,100 USD. Clusters 5 and 9 involve transactions with larger amounts, with an average SrcAmt of $15,000 USD. Also, normal transactions have low transaction frequencies; in clusters 0, 3, and 5, 90% of transaction graphs have no more than five transactions, while in clusters 7 and 9, 90% have no more than 15 transactions. Their average transfer loss is 2.43%. From the perspective of transaction networks, these transactions are relatively simple, with simple network structures and low network-related feature values. Overall, these low-frequency users primarily use cross-chain bridges for asset transfers.

G2: Large Transfers. G2 represents large-scale cross-chain fund transfers, consisting of 4 groups, 8 addresses, and 138 transactions. These transactions have an average SrcAmt of $31,526, with a significantly higher transfer loss (5.58%) compared to normal transactions (G1). This aligns with our findings in §5.3, where high *WkFreqMax* indicates frequent cross-chain transfers over short periods, reducing pool liquidity and increasing transfer costs. From a graph perspective, G2's DegGini and Density are both 0, indicating a sparse transaction structure. Analysis reveals that G2 includes continuous transfers and self-loops (i.e., SrcAddr and DstAddr are the same), leading to a low DegGini. Compared to arbitrage bots (G3), G2 has fewer transactions and lower interaction frequency, resulting in a lower normalized Density.

Notably, none of these addresses have been flagged as phishing or attacker addresses. We speculate they are controlled by institutions or large entities with low initial holding costs, making them less sensitive to transfer losses. Additionally, these addresses frequently participate in DeFi activities, suggesting that their cross-chain transactions primarily serve investment purposes, such as providing liquidity to trading pools or engaging in other DeFi strategies.

G3: Arbitrage Bots. Liquidity pool-based bridges, like Allbridge, offer arbitrage opportunities as token values fluctuate with supply and demand across different chains. Arbitrage bots exploit these price differences for profit by executing rapid cross-chain transactions. We identified 11 arbitrage bots that collectively profited $267,850 through cross-chain arbitrage. These bots exhibit distinctive features, as shown in Table 10.

- Cluster 1 (1 bot, 6 addrs, 4,660 txs) represents large-scale arbitrage. The bot executed 4,386 profitable transactions (94.12%), earning $121,337. It appears as a significant outlier (yellow hexagon) in Fig. 8a. The bot has high-frequency transactions, with DegGini (1.0), DegAvg (1.0), and Density (1.0). Most transactions are concentrated in a few key addresses, while others handle fund transfers or profit settlement.
- Cluster 8 (3 bots, 28 addrs, 2,597 txs) represents medium-scale arbitrage. The larger number of addresses results in lower DegAvg (0.1) and Density (0.1) than Cluster 1. However, DegGini (1.0) indicates a similar structure to cluster 1, where key addresses dominate while others are loosely connected.
- Cluster 6 (7 bots, 15 addrs, 2,106 txs) represents a mix of small- and medium-scale arbitrage. The lower DegGini (0.3) suggests a more even transaction distribution. Density (0.8) remains relatively high due to a few dominant bots, but overall, the structure is more balanced than Clusters 1 and 8.

Despite some transfer losses, these bots remain highly profitable. For instance, Cluster 1 had a cumulative loss of $1,910, yet secured over $100,000 in net profit. Arbitrage bots play a dual role in the market. On one hand, they increase liquidity by frequently trading assets across chains. On the other hand, their high-speed execution gives them an advantage over normal users, limiting opportunities for others to profit from price discrepancies.

G4: Liquidity Pool (LP) Attack. In Fig. 8a, there is another significant outlier (the red square) belonging to cluster 2, recognized by Allbridge officials as an attack [51]. On April 2, 2023, Allbridge experienced a flash loan exploit on the BSC. The stablecoin pools for USDT and BUSD were attacked, resulting in hackers stealing approximately $570,000 USD. The root cause was a logical flaw in the

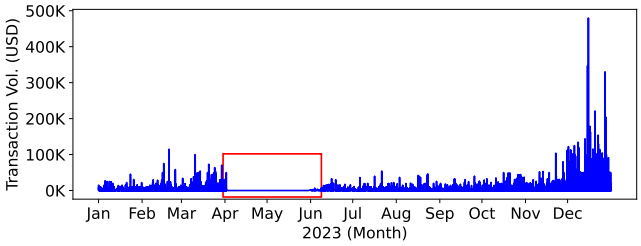


Fig. 9: Allbridge Daily Transaction Volume in 2023

cross-chain contract; the attacker acted as both a liquidity provider and swapper, enabling them to manipulate prices and drain funds from the pool. The attacker used a single transaction to flash loan $7.5 million BUSD, then manipulated the liquidity of the pool to affect token values, ultimately gaining $570,000 in profits.

From Table 10, the LP attack differs from other transactions. First, the attacker can obtain more funds through flash loans, so its *SrcAmtMax* is not very high. Second, its *ProfitSum* is significantly higher than others, including arbitrage transactions. The attacker's transaction graph is simple, involving only one transaction with no other cross-chain activity. The attacker targeted the liquidity pool, conducting a single on-chain transaction where both the SrcChain and DstChain were on BSC, explaining the zero time cost in §5.1. This attack not only caused financial losses but also led to the suspension of the cross-chain bridge service. Fig. 9 tracks Allbridge's daily transaction volume. After the attack was announced on April 2, the transaction volume dropped to zero and only began recovering at the end of May, about two months later.

Finding 6. We conducted a cluster analysis of transaction graphs and identified four main types of activities: *normal transactions, large transfers, arbitrage bots*, and *liquidity pool attacks*. Large transfers caused significant market fluctuations, increasing transfer costs. Arbitrage bots accumulated substantial profits within a few months, and a single LP attack resulted in over $570,000 in losses and a two-month service suspension. These abnormal activities strained regular users by reducing liquidity and driving up costs.

7.4 Malicious Address Activities

This section investigates whether known malicious addresses are involved in cross-chain transactions. Metasleuth.io, a website that tracks malicious addresses, has reported 1574 malicious address related to attacks on Ethereum transactions. We collected these malicious addresses and tracked their cross-chain activities.

We first checked if the *fromAddress* and *toAddress* in cross-chain transactions matched any reported malicious addresses. Using the identified transactions, we traced all related cross-chain activities through transaction graphs in §7.2. Finally, we discovered 24 malicious addresses involved in 82 cross-chain transactions in 24 transaction graphs, with a cumulative transfer amount of $186,043. All these transactions fell under *G1: Normal Transactions*, indicating that asset transfers were the main purpose of these malicious addresses' cross-chain activities. We also identified one *Pink Drainer address* involved in 26 cross-chain transactions. As reported by Metasleuth.io, the Pink Drainer organizations were involved in 159 phishing incidents, resulting in losses exceeding $5.44 million. Notably, 91.46% (75 transactions) of these cross-chain transactions were transferred out from the Binance Smart Chain (BSC), suggesting that BSC was a frequent target.

Table 11: Newly Identified Potentially Malicious Addresses

#	Blockchain	Address
1	Binance (BSC)	0x2b8ea90eae79c632c05f5ff39fbc337145cadcca
2	TRON (TRX)	ta9qhksusxrwxqvvra5rim7dlt8ih94rd4
3	TRON (TRX)	tuidc4msrenvohaa63ddtpzz9yeycfdgax
4	Avalanche (AVA)	0xb7394b66f64d266fc6c64b9a73648aa462064cfb
5	Avalanche (AVA)	0x9b2cc969615ca71b190fbf3e881b90eb97e13ee2
6	Polygon (POL)	0x8afea410deeaa58adba6af384ae5e9309b49d8f2

Based on the 24 transaction graphs, we also found six previously unreported malicious addresses, as shown in Table 11. We have reported these addresses to relevant blockchain explorers. Traditional methods for tracking malicious addresses are limited to blockchains that use the same address format. For example, if an address is confirmed to be involved in an attack on Ethereum (ETH), its activities can only be tracked on blockchains with a matching format. However, if the two blockchains use different address formats, as is the case with BSC and TRX in Table 11, this method becomes ineffective. By analyzing cross-chain transactions, we can overcome these limitations and trace malicious activities across blockchains with different address formats.

Finding 7. We identified 24 reported malicious addresses involved in 82 cross-chain transactions, totaling $186,043. The primary activity of these addresses on cross-chain bridges was the transfer of illicit funds. Additionally, we uncovered six previously unreported suspicious addresses.

8 Limitations and Discussion

8.1 Limitations

Our study have several limitations. First, we only focus on cross-chain bridges with official APIs, which only covers a subset of all cross-chain bridges. Moreover, we excluded cross-chain transactions that have incomplete information. Second, we only analyzed cross-chain transfers with the standard Transfer function. About 2% of on-chain transactions do not use this function and may involve custom transfer functions, which we did not consider. Future work can study these customized transfer functions. Third, in time cost analysis, we use block timestamps to calculate the time cost, which only representing the lower bound of the actual time cost. Fourth, for arbitrage analysis, we did not consider on-chain gas fees. Besides, another common arbitrage strategy involves taking advantage of price discrepancies between chains, but we did not investigate this due to the lack of comprehensive historical price data across different blockchains. We leave a complete study to address these limitations as our future work.

8.2 Discussion

Collecting and analyzing cross-chain transaction data is challenging due to the lack of public APIs, inconsistent data, and limited transaction details from many cross-chain bridges. To address this, we supplemented our analysis with additional blockchain data. These issues stem from the lack of standardization in cross-chain bridges, which we discuss next.

Standardization. Our study reveals that most cross-chain bridges lack standardized ledger structures and smart contracts. We identified 20,676 on-chain transactions with non-standard events across the four bridges, highlighting inconsistencies in cross-chain contract implementations. The lack of standardization can lead to asset losses. For example, we found failed on-chain transactions that were not marked as exceptions in the Connext ledger, suggesting inadequate standards for exception handling and transaction tracking. Additionally, bridges are also vulnerable to LP attacks, like the Allbridge exploit on April 2, 2023, where flash loan manipulation caused a $570k loss and a temporary shutdown. These issues highlight the need for robust mechanisms in cross-chain bridges.

Recommendations. Based on our study, we provide the following recommendations for bridge developers.

For Existing Bridges: Enhancing transparency is crucial to mitigating ledger inconsistencies. Cross-chain bridges should: 1) Clearly indicate asset unit conversions and provide asset mapping details in their APIs to help developers correctly handle unit differences. 2) Continuously monitor and accurately interpret on-chain transactions to detect and resolve anomalies in a timely manner. 3) Conduct security audits on smart contracts and implement real-time monitoring of on-chain states to prevent attacks.

For Future Bridges: New cross-chain bridge designs should adopt standardized frameworks to prevent ecosystem fragmentation and improve interoperability. Standardization should include: 1) Unified APIs that follow a structured format for asset transfers, making integration smoother for developers. 2) Ensuring consistency between on-chain and off-chain data to avoid conversion mismatches and precision discrepancies. 3) Comprehensive transaction logging and standardized exception handling, ensuring complete and transparent ledger records.

Future Work. Transaction graphs provide an innovative viewpoint for detecting hidden cross-chain activities from cross-chain transactions. Future research could delve more deeply into specific actions such as substantial transfers and arbitrage bots. Additionally, monitoring and examining malicious addresses within cross-chain transactions presents a valuable area for further investigation.

9 Related Work

To the best of our knowledge, our work is the first to systematically analyze cross-chain transactions by analyzing both cross-chain bridge and blockchain data. Our empirical analysis provides insights into the performance and security of cross-chain bridges, shedding light on potential risks and vulnerabilities.

Cross-chain technology. enables secure and efficient communication between different blockchains, facilitating asset transfers [25, 38, 49] and message passing [35, 59, 74]. Common cross-chain technologies include cross-chain bridges [55–57, 67], relays [63, 72], sidechains [12], and Hash Time-Locked Contracts (HTLCs) [11, 24, 28, 59]. Ou et al. [49] provided an overview of cross-chain technology, including mechanisms, platforms, challenges, and advances. They discussed the main cross-chain technologies, such as interoperability, trust model, transaction speed and security. Lin et al. [38] provides a systematic overview of cross-chain asset transfer schemes, introducing their classification, main challenges, and representative implementations. Zamyatin et al. [74] present a general framework to design and evaluate cross-chain communication protocols that facilitate blockchain interoperability. Existing works mainly focus on specific cross-chain technologies, while our work abstracts the cross-chain transaction process to systematically measure and analyze the performance and security of different cross-chain bridges.

Cross-chain bridge security. Existing works [26, 68] have analyzed issues such as security and privacy of cross-chain bridges from different perspectives. Zhang et al. [76] performed a systematic study of cross-chain bridge security issues, identified 12 potential attack vectors. Lee et al. [32] analyzed bridge attacks from four components and reviewed 8 exploits from real-world and discussed the mitigation. Xscope [75] is an automatic tool to find security violations in cross-chain bridges. It uses a set of security properties and patterns to detect security issues in cross-chain bridges. Han et al. [25] conducted a survey on blockchain interoperability, focusing on security and privacy challenges. Compared to the above works, our work provides a comprehensive empirical analysis of cross-chain bridges from both bridge and blockchain transaction data.

Blockchain measurement. Many studies empirically analyze various aspects of blockchain, such as transactions and interactions among entities [3, 10, 30, 33, 37, 42, 43, 53, 61, 70, 78]. Some focus on the effects of specific protocols post-Ethereum upgrade, like EIP-1559 [34, 39, 52], and on decentralized applications (DApps) like Uniswap [27]. Miner Extractable Value (MEV) or Block Extractable Value (BEV) have also been studied extensively [6, 16, 40, 50, 60, 62, 71, 77]. These works provide valuable insights into the performance and security of blockchains, but they do not focus on cross-chain transactions and bridges.

10 Conclusion

This paper combines bridge transaction data and on-chain transaction data to present the first large-scale measurement study of four cross-chain bridges. Our work focuses on cost metrics, inconsistencies, and activities in cross-chain transactions. Overall, we find that while performance varies across the four bridges, most regular transactions are completed within minutes and at low costs, typically just a few dollars, making them practical. However, we also identified issues such as transactions with unusually high costs and inconsistencies between bridge and blockchain ledgers. Additionally, we uncover various activities, including liquidity pool attacks that have caused significant losses to bridges, as well as large transfers and arbitrage bots.

Acknowledgments

This project was supported in part by an academic grant from Ethereum Foundation. The views and conclusions contained in this document are those of the authors only.

References

[1] Allbridge. 2024. Allbridge Documentation. https://docs-core.allbridge.io/product/how-does-allbridge-core-work/messaging-protocols.

[2] Allbridge. 2024. Allbridge official website. https://allbridge.io.

[3] Qianlan Bai, Chao Zhang, Nianyi Liu, Xiaowei Chen, Yuedong Xu, and Xin Wang. 2022. Evolution of Transaction Pattern in Ethereum: A Temporal Graph Perspective. *IEEE Trans. Comput. Soc. Syst.* 9, 3 (2022), 851–866. https://doi.org/10.1109/TCSS.2021.3108788

[4] bitpay. 2022. What is a Crypto Swap and How Do I Start Swapping. https://bitpay.com/blog/what-is-a-crypto-swap/.

[5] Vitalik Buterin. 2024. A Next-Generation Smart Contract and Decentralized Application Platform. https://ethereum.org/en/whitepaper/.

[6] Agostino Capponi, Ruizhe Jia, and Ye Wang. 2022. The Evolution of Blockchain: from Lit to Dark. *arXiv preprint* (2022).

[7] Chainlink. 2022. 77+ Smart Contract Use Cases Enabled By Chainlink. https://blog.chain.link/smart-contract-use-cases/.

[8] Chainlink. 2023. Chainlink: Blockchain oracles for hybrid smart contracts. https://chain.link/cross-chain.

[9] Chainlink. 2024. Understanding Cross-Chain Token Transfers. https://chain.link/education-hub/cross-chain-token-transfers.

[10] Ting Chen, Zihao Li, Yuxiao Zhu, Jiachi Chen, Xiapu Luo, John Chi-Shing Lui, Xiaodong Lin, and Xiaosong Zhang. 2020. Understanding Ethereum via Graph Analysis. *ACM Trans. Internet Techn.* 20, 2 (2020), 18:1–18:32. https://doi.org/10.1145/3381036

[11] Yulong Chen, Alia Asheralieva, and Xuetao Wei. 2024. AtomCI: A New System for the Atomic Cross-Chain Smart Contract Invocation Spanning Heterogeneous Blockchains. *IEEE Trans. Netw. Sci. Eng.* (2024).

[12] João Otávio Massari Chervinski, Diego Kreutz, and Jiangshan Yu. 2023. Towards Scalable Cross-Chain Messaging. In *Proceedings of IEEE International Conference on Blockchain, Blockchain 2023.*

[13] Coinbase. 2024. What is DeFi. https://www.coinbase.com/learn/crypto-basics/plp-what-is-defi.

[14] Connext. 2024. Connext Documentation. https://docs.connext.network/.

[15] Connext. 2024. Connext official website. https://www.connext.network.

[16] Philip Daian, Steven Goldfeder, Tyler Kell, Yunqi Li, Xueyuan Zhao, Iddo Bentov, Lorenz Breidenbach, and Ari Juels. 2020. Flash Boys 2.0: Frontrunning in Decentralized Exchanges, Miner Extractable Value, and Consensus Instability. In *Proceedings of 2020 IEEE Symposium on Security and Privacy, SP 2020*. IEEE, 910–927. https://doi.org/10.1109/SP40000.2020.00040

[17] DappRadar. 2024. State of the Dapp Industry Q1 2024. https://dappradar.com/blog/state-of-the-dapp-industry-q1-2024.

[18] DappRadar. 2024. Top Blockchains by Total Value Locked (TVL). https://dappradar.com/rankings/chains?sort=tvlInFiat&order=desc.

[19] Dipanjan Das, Priyanka Bose, Nicola Ruaro, Christopher Kruegel, and Giovanni Vigna. 2022. Understanding Security Issues in the NFT Ecosystem. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, CCS 2022.*

[20] Defillama. 2024. Bridge total value hacked. https://defillama.com/hacks.

[21] Defillama. 2024. Bridge TVL Rankings. https://defillama.com/protocols/Bridge.

[22] Ethereum. 2024. ERC-20 TOKEN STANDARD. https://ethereum.org/en/developers/docs/standards/tokens/erc-20/.

[23] Etherscan. 2024. Etherscan API Documentation. https://docs.etherscan.io/api-endpoints/geth-parity-proxy.

[24] Yihao Guo, Minghui Xu, Dongxiao Yu, Yong Yu, Rajiv Ranjan, and Xiuzhen Cheng. 2023. Cross-Channel: Scalable Off-Chain Channels Supporting Fair and Atomic Cross-Chain Operations. *IEEE Trans. Computers* (2023).

[25] Panpan Han, Zheng Yan, Wenxiu Ding, Shufan Fei, and Zhiguo Wan. 2023. A Survey on Cross-chain Technologies. *Distributed Ledger Technol. Res. Pract.* (2023).

[26] Terje Haugum, Bjørnar Hoff, Mohammed Alsadi, and Jingyue Li. 2022. Security and Privacy Challenges in Blockchain Interoperability - A Multivocal Literature Review. In *Proceedings of the International Conference on Evaluation and Assessment in Software Engineering, EASE 2022.*

[27] Lioba Heimbach, Eric Schertenleib, and Roger Wattenhofer. 2022. Risks and Returns of Uniswap V3 Liquidity Providers. In *Proceedings of the 4th ACM Conference on Advances in Financial Technologies, AFT 2022*. ACM, 89–101. https://doi.org/10.1145/3558535.3559772

[28] Maurice Herlihy. 2018. Atomic Cross-Chain Swaps. In *Proceedings of the ACM the 2018 ACM Symposium on Principles of Distributed Computing, PODC 2018*, Calvin Newport and Idit Keidar (Eds.). ACM, 245–254.

[29] Soichiro Imoto, Yuichi Sudo, Hirotsugu Kakugawa, and Toshimitsu Masuzawa. 2023. Atomic cross-chain swaps with improved space, time and local time complexities. *Inf. Comput.* (2023).

[30] Lucianna Kiffer, Dave Levin, and Alan Mislove. 2018. Analyzing Ethereum's Contract Topology. In *Proceedings of the Internet Measurement Conference 2018, IMC 2018.*

[31] Oliver Knight. 2024. Orbit Chain Loses $81M in Cross-Chain Bridge Exploit. https://www.coindesk.com/business/2024/01/02/orbit-chain-loses-81m-in-cross-chain-bridge-exploit/.

[32] Sung-Shine Lee, Alexandr Murashkin, Martin Derka, and Jan Gorzny. 2023. SoK: Not Quite Water Under the Bridge: Review of Cross-Chain Bridge Hacks. In *Proceedings of IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2023.*

[33] Xi Tong Lee, Arijit Khan, Sourav Sen Gupta, Yu Hann Ong, and Xuan Liu. 2020. Measurements, Analyses, and Insights on the Entire Ethereum Blockchain Network. In *Proceedings of the Web Conference 2020, WWW 2020.*

[34] Stefanos Leonardos, Barnabé Monnot, Daniël Reijsbergen, Efstratios Skoulakis, and Georgios Piliouras. 2021. Dynamical analysis of the EIP-1559 Ethereum fee market. In *Proceedings of 3rd ACM Conference on Advances in Financial Technologies, AFT 2021.*

[35] Jiasun Li and Zhengxun Wu. 2023. Arbitrary message passing across blockchains. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4417670.

[36] Dan Lin, Jiajing Wu, Yuxin Su, Ziye Zheng, Yuhong Nan, and Zibin Zheng. 2024. CONNECTOR: Enhancing the Traceability of Decentralized Bridge Applications via Automatic Cross-chain Transaction Association. *arXiv preprint* (2024).

[37] Dan Lin, Jiajing Wu, Qi Yuan, and Zibin Zheng. 2020. Modeling and Understanding Ethereum Transaction Records via a Complex Network Approach. *IEEE Trans. Circuits Syst.* 67-II, 11 (2020), 2737–2741. https://doi.org/10.1109/TCSII.2020.2968376

[38] Lu Lin, Jiayi Li, Yuzhen Wang, and Qiong Wang. 2023. A Survey on Cross-Chain Asset Transfer Schemes: Classification, Challenges, and Prospects. In *Proceedings of International Conference on Networking and Network Applications, NaNA 2023.*

[39] Yulin Liu, Yuxuan Lu, Kartik Nayak, Fan Zhang, Luyao Zhang, and Yinhong Zhao. 2022. Empirical Analysis of EIP-1559: Transaction Fees, Waiting Times, and Consensus Security. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022.*

[40] Xingyu Lyu, Mengya Zhang, Xiaokuan Zhang, Jianyu Niu, Yinqian Zhang, and Zhiqiang Lin. 2022. An Empirical Study on Ethereum Private Transactions and the Security Implications. *arXiv preprint* (2022).

[41] Yakko Majuri. 2018. Simply Explained: Ethereum Gas. https://yakkomajuri.medium.com/blockchain-definition-of-the-week-ethereum-gas-2f976af774ed.

[42] Juliana Zanelatto Gavião Mascarenhas, Artur Ziviani, Klaus Wehmuth, and Alex Borges Vieira. 2020. On the transaction dynamics of the Ethereum-based cryptocurrency. *J. Complex Networks* 8, 4 (2020). https://doi.org/10.1093/COMNET/CNAA042

[43] Ori Mazor and Ori Rottenstreich. 2024. An Empirical Study of Cross-Chain Arbitrage in Decentralized Exchanges. In *Proceedings of 16th International Conference on COMmunication Systems & NETworkS, COMSNETS 2024.*

[44] Binance News. 2024. Crypto Scam Kingpin 'Pink Drainer' Announces Retirement After Stealing Millions. https://www.binance.com/en/square/post/2024-05-17-crypto-scam-kingpin-pink-drainer-announces-retirement-after-stealing-millions-8240743529769.

[45] OKX. 2024. OKX Documentation. https://www.okx.com/web3/build/docs/waas/dex-crosschain-faq.

[46] Openzeppelin. 2024. Token introduction. https://docs.openzeppelin.com/contracts/5.x/tokens.

[47] Orbit. 2024. Orbit Documentation. https://docs.orbitchain.io/.

[48] Orbit. 2024. Orbit official website. https://bridge.orbitchain.io/.

[49] Wei Ou, Shiying Huang, Jingjing Zheng, Qionglu Zhang, Guang Zeng, and Wenbao Han. 2022. An overview on cross-chain: Mechanism, platforms, challenges and advances. *Comput. Networks* (2022).

[50] Julien Piet, Jaiden Fairoze, and Nicholas Weaver. 2022. Extracting Godl [sic] from the Salt Mines: Ethereum Miners Extracting Value. *arXiv preprint* (2022).

[51] QuillAudits. 2023. Decoding AllBridge's $570K Flash Loan Exploit. https://medium.com/coinmonks/decoding-allbridge-570k-flash-loan-exploit-quillaudits-8da8dccd729d.

[52] Daniël Reijsbergen, Shyam Sridhar, Barnabé Monnot, Stefanos Leonardos, Stratis Skoulakis, and Georgios Piliouras. 2021. Transaction Fees on a Honeymoon: Ethereum's EIP-1559 One Month Later. In *Proceedings of 2021 IEEE International Conference on Blockchain, Blockchain 2021*. IEEE, 196–204. https://doi.org/10.1109/BLOCKCHAIN53845.2021.00034

[53] Anwar Said, Muhammad Umar Janjua, Saeed-Ul Hassan, Zeeshan Muzammal, Tania Saleem, Tipajin Thaipisutikul, Suppawong Tuarob, and Raheel Nawaz. 2021. Detailed analysis of Ethereum network on transaction behavior, community structure and link prediction. *PeerJ Comput. Sci.* 7 (2021), e815. https://doi.org/10.7717/PEERJ-CS.815

[54] Narges Shadab, Farzin Houshmand, and Mohsen Lesani. 2020. Cross-chain Transactions. In *Proceedings of IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2020.*

[55] Peiyao Sheng, Xuechao Wang, Sreeram Kannan, Kartik Nayak, and Pramod Viswanath. 2023. TrustBoost: Boosting Trust among Interoperable Blockchains. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023.*

[56] Hong Su, Bing Guo, Jun Yu Lu, and Xinhua Suo. 2022. Cross-chain exchange by transaction dependence with conditional transaction method. *Soft Comput.* (2022).
[57] Erkan Tairi, Pedro Moreno-Sanchez, and Clara Schneidewind. 2023. LedgerLocks: A Security Framework for Blockchain Protocols Based on Adaptor Signatures. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023.*
[58] Techdreams. 2024. Sushiswap vs Uniswap Gas Fees On Ethereum. https://www.techdreams.org/crypto-currency/sushiswap-vs-uniswap-gas-fees-on-ethereum/12576-20220131.
[59] Sri Aravinda Krishnan Thyagarajan, Giulio Malavolta, and Pedro Moreno-Sanchez. 2022. Universal Atomic Swaps: Secure Exchange of Coins Across All Blockchains. In *Proceedings of 43rd IEEE Symposium on Security and Privacy, SP 2022.*
[60] Christof Ferreira Torres, Ramiro Camino, and Radu State. 2021. Frontrunner Jones and the Raiders of the Dark Forest: An Empirical Study of Frontrunning on the Ethereum Blockchain. In *Proceedings of 30th USENIX Security Symposium, USENIX Security 2021.* USENIX Association, 1343–1359. https://www.usenix.org/conference/usenixsecurity21/presentation/torres
[61] Natkamon Tovanich, Nicolas Soulié, Nicolas Heulot, and Petra Isenberg. 2021. An Empirical Analysis of Pool Hopping Behavior in the Bitcoin Blockchain. In *Proceedings of IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2021.*
[62] Ben Weintraub, Christof Ferreira Torres, Cristina Nita-Rotaru, and Radu State. 2022. A flash(bot) in the pan: measuring maximal extractable value in private pools. In *Proceedings of the 22nd ACM Internet Measurement Conference, IMC 2022,* Chadi Barakat, Cristel Pelsser, Theophilus A. Benson, and David R. Choffnes (Eds.). ACM, 458–471. https://doi.org/10.1145/3517745.3561448
[63] Martin Westerkamp and Maximilian Diez. 2022. Verilay: A Verifiable Proof of Stake Chain Relay. In *Proceedings of IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2022.*
[64] Bryan White, Aniket Mahanti, and Kalpdrum Passi. 2022. Characterizing the OpenSea NFT Marketplace. In *Proceedings of the Companion of The Web Conference, WWW Companion 2022.*
[65] Wormhole. 2024. Wormhole Documentation. https://wormhole.com/docs/learn/infrastructure/relayer/.
[66] Wormhole. 2024. Wormhole official website. https://wormhole.com/.
[67] Tiancheng Xie, Jiaheng Zhang, Zerui Cheng, Fan Zhang, Yupeng Zhang, Yongzheng Jia, Dan Boneh, and Dawn Song. 2022. zkBridge: Trustless Cross-chain Bridges Made Practical. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022.*
[68] Minghui Xu, Yihao Guo, Chun-Chi Liu, Qin Hu, Dongxiao Yu, Zehui Xiong, Dusit Niyato, and Xiuzhen Cheng. 2023. Exploring Blockchain Technology through a Modular Lens: A Survey. *CoRR* (2023).
[69] Kailun Yan, Jilian Zhang, Xiangyu Liu, Wenrui Diao, and Shanqing Guo. 2023. Bad Apples: Understanding the Centralized Security Risks in Decentralized Ecosystems. In *Proceedings of the ACM Web Conference 2023, WWW 2023.* ACM, 2274–2283. https://doi.org/10.1145/3543507.3583393
[70] Kailun Yan, Xiaokuan Zhang, and Wenrui Diao. 2024. Stealing Trust: Unraveling Blind Message Attacks in Web3 Authentication. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security, CCS 2024.*
[71] Sen Yang, Fan Zhang, Ken Huang, Xi Chen, Youwei Yang, and Feng Zhu. 2022. SoK: MEV Countermeasures: Theory and Practice. *arXiv preprint* (2022).
[72] Lingyuan Yin, Jing Xu, and Qiang Tang. 2022. Sidechains With Fast Cross-Chain Transfers. *IEEE Trans. Dependable Secur. Comput.* (2022).
[73] Haaroon Yousaf, George Kappos, and Sarah Meiklejohn. 2019. Tracing Transactions Across Cryptocurrency Ledgers. In *Proceedings of 28th USENIX Security Symposium, USENIX Security 2019.*
[74] Alexei Zamyatin, Mustafa Al-Bassam, Dionysis Zindros, Eleftherios Kokoris-Kogias, Pedro Moreno-Sanchez, Aggelos Kiayias, and William J. Knottenbelt. 2021. SoK: Communication Across Distributed Ledgers. In *Proceedings of Financial Cryptography and Data Security - 25th International Conference, FC 2021.*
[75] Jiashuo Zhang, Jianbo Gao, Yue Li, Ziming Chen, Zhi Guan, and Zhong Chen. 2022. Xscope: Hunting for Cross-Chain Bridge Attacks. In *Proceedings of 37th IEEE/ACM International Conference on Automated Software Engineering,ASE 2022.*
[76] Mengya Zhang, Xiaokuan Zhang, Josh Barbee, Yinqian Zhang, and Zhiqiang Lin. 2024. Security of Cross-chain Bridges: Attack Surfaces, Defenses, and Open Problems. In *Proceedings of the 27th International Symposium on Research in Attacks, Intrusions and Defenses, RAID 2024.*
[77] Wuqi Zhang, Lili Wei, Shing-Chi Cheung, Yepang Liu, Shuqing Li, Lu Liu, and Michael R. Lyu. 2023. Combatting Front-Running in Smart Contracts: Attack Mining, Benchmark Construction and Vulnerability Detector Evaluation. *IEEE Trans. Software Eng.* 49, 6 (2023), 3630–3646. https://doi.org/10.1109/TSE.2023.3270117
[78] Lin Zhao, Sourav Sen Gupta, Arijit Khan, and Robby Luo. 2021. Temporal Analysis of the Entire Ethereum Blockchain Network. In *Proceedings of the Web Conference 2021, WWW 2021.*

A Formalized Definitions

Based on the cross-chain workflow shown in Fig. 1, we provide the formalized definitions of cross-chain transactions in Table 12. These definitions outline the key actions and their corresponding descriptions, providing a systematic view of the cross-chain process.

Table 12: Formal Definition of Cross-chain Transaction

#	Action	Description
❶	Send SrcTx	$\mathbf{A}_s \xrightarrow{\mathbf{tx}_s} \mathbf{C}_s : tx_s = (\mathbf{A}_s, \mathbf{sc}_s, amt_s)$
❷	Lock/Burn Asset	$\mathbf{tx}_s \xrightarrow{call} \mathbf{sc}_s$: Lock/Burn $(\mathbf{A}_s, amt_s)$
❸	Emit *event*	$\mathbf{sc}_s \xrightarrow{\text{emit}} \mathbf{C}_s : event_s = (\mathbf{A}_s, \mathbf{sc}_s, amt_s)$
❹	Update Record	$\mathbf{B} \xleftarrow{event_s} \mathbf{C}_s : br \leftarrow (\mathbf{A}_s, \mathbf{sc}_s, amt_s, \mathcal{H}(tx_s))$
❺	Send DstTx	$\mathbf{B} \xrightarrow{\mathbf{tx}_d} \mathbf{C}_d : tx_d = (\mathbf{A}_d, \mathbf{sc}_d, amt_d)$
❻	Mint/Release Asset	$\mathbf{tx}_d \xrightarrow{call} \mathbf{sc}_d$: Unlock/Mint $(\mathbf{A}_d, amt_d)$
❼	Emit *event*	$\mathbf{sc}_d \xrightarrow{\text{emit}} \mathbf{C}_d : event_d = (\mathbf{A}_d, \mathbf{sc}_d, amt_d)$
❽	Update Record	$\mathbf{B} \xleftarrow{event_d} \mathbf{C}_d : br \leftarrow (\mathbf{A}_d, \mathbf{sc}_d, amt_d, \mathcal{H}(tx_d))$

B Cross-chain Transaction Dataset

B.1 Cross Chain Bridges Selection

Table 13 lists the 30 bridges we considered. We used two APIs for data collection: the *Tx Info API* and the *Tx Index API*. The *Tx Info API* provides details of individual cross-chain transactions, while the *Tx Index API* offers an index of transactions over a specified period, such as a set of transaction hashes. Some bridges only offer the *Tx Info API*, which requires a transaction hash to fetch details. Without the *Tx Index API*, we cannot retrieve all transactions within a specific time frame (e.g., a year). Only four bridges provide both APIs, so we selected them for our analysis. The *Chains Supported* column shows the chains each bridge claims to support, according to their API documentation. However, these claims may not reflect the actual data. For example, Allbridge claims to support 21 chains, but we only collected data from 8 chains in 2023.

B.2 Transfer Events

Transfer Events. The Transfer function is central to cross-chain transactions, involving the movement of tokens. In blockchain ledger, *transaction logs (receipts)* record all events emitted by cross-chain contracts, which we can obtain by their event signature. Typically, the SrcTx includes a TransferEvent recording the sender address (SrcAddr) that transferred tokens (SrcAmt), while the DstTx includes a TransferEvent recording the cross-chain bridge transferring tokens (DstAmt) to the receiver address (DstAddr). We conducted a preliminary analysis of the TransferEvents in on-chain transactions. Specifically, we matched the SrcAddr and DstAddr provided in the bridge ledger with the TransferEvents in SrcTx and DstTx, respectively. We found that 16,901 SrcTx and 4,775 DstTx lacked transfer events, accounting for 2.01% of the total on-chain transactions.

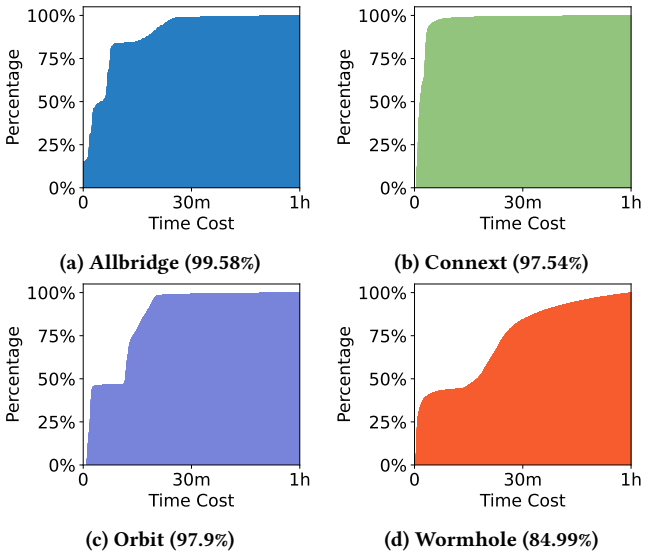
C CDF of Time Cost

In this section, we present the cumulative distribution function (CDF) of the time cost for four bridges. As most of the time costs are relatively small, we limit the visualization to one hour for clarity. Fig. 10 shows that, for most bridges, the cumulative time cost approaches 100% around the 30 minutes, except for Wormhole.

Table 13: Cross-Chain Bridge API Support Overview

#	Bridge	Layer	Chains Supported	Tx Info API	Tx Index API
1	**PolygonBridge**	L1-L2	ETH, POL	✗	✗
2	**ArbitrumBridge**	L1-L2	ETH, ARB	✗	✗
3	**RainbowBridge**	L1-L1	ETH, Near	✗	✗
4	**xDAIBridgex**	L1-L2	ETH, Gnosis	✗	✗
5	**WrapProtocol**	L1-L1	ETH, Tezzo	✗	✗
6	**AvalancheBridge**	L1-L1	ETH, AVAX	✗	✗
7	**RenBridge**	L1-L1	ETH, Bitcoin	✔	✗
8	**RSKTokenBridge**	L1-L2	ETH, RSK	✗	✗
9	**SovrynBridge**	L1-L2	ETH, RSK, BNB	✗	✗
10	**VoltageBridge**	L1-L2	ETH, Fuse, BNB	✗	✗
11	**Hot Cross**	L1-L2	ETH, AVAX, BNB	✗	✗
12	**CeloOpticsBridge**	L1-L2	ETH, POL, Celo	✗	✗
13	**ioTubeBridge**	L1-L2	4 chains	✔	✗
14	**Nomad**	L1-L2	4 chains	✗	✗
15	**ThunderCoreBridge**	L1-L2	4 chains	✗	✗
16	**Cross-ChainBridge**	L1-L2	5 chains	✗	✗
17	**AcrossProtocol**	L1-L2	5 chains	✗	✗
18	**SOYBridge**	L1-L2	5 chains	✔	✗
19	**Connext**	L1-L2	7 chains	✔	✔
20	**HyphenBridge**	L1-L2	7 chains	✔	✗
21	**SatellitebyAxelar**	L1-L2	14 chains	✗	✗
22	**BoringDAOBridge**	L1-L2	14 chains	✗	✗
23	**ChainPortBridge**	L1-L2	17 chains	✗	✗
24	**SynapseProtocol**	L1-L2	19 chains	✔	✗
25	**Orbit**	L1-L2	21 chains	✔	✔
26	**Allbridge**	L1-L2	21 chains	✔	✔
27	**OptimismBridge**	L1-L2	24 chains	✗	✗
28	**Multichain**	L1-L2	26 chains	✔	✗
29	**Wormhole**	L1-L2	30 chains	✔	✔
30	**CelercBridge**	L1-L2	34 chains	✔	✗

L1-L1: Cross-chain transactions between different Layer 1 blockchains;
L1-L2: Cross-chain transactions between Layer 1 and Layer 2 networks.



(a) Allbridge (99.58%) (b) Connext (97.54%)

(c) Orbit (97.9%) (d) Wormhole (84.99%)

Fig. 10: CDF of Time Cost (≤ 1 Hours)

Algorithm 1: Inconsistency Detection

Function `LogFloor`(x):
  Input: x - the input number
  Output: r - the count of powers of 10
  $r = 0$;
  while $x \geq 10$ **do**
    $x = \lfloor \frac{x}{10} \rfloor$;
    $r = r + 1$;
  return r;

Function `Inconsistency Detection`$(amt_s, amt_d, amt'_s, amt'_d)$:
  Input: $amt_s, amt_d, amt'_s, amt'_d$ - amounts in ledger and on-chain transcations
  Output: Inconsistencies - a list of detected inconsistencies
  AmtInconsistency = False;
  Type1Inconsistency = False;
  Type2Inconsistency = False;
  if $amt_s \neq amt'_s$ *or* $amt_d \neq amt'_d$ **then**
    AmtInconsistency = True;
  s_1 = LogFloor(amt_s);
  d_1 = LogFloor(amt_d);
  s_2 = LogFloor$(amt_s{}')$;
  d_2 = LogFloor$(amt_d{}')$;
  if $s_1 \neq s_2$ *or* $d_1 \neq d_2$ **then**
    Type1Inconsistency = True;
  if $s_1 \neq d_1$ *or* $s_2 \neq d_2$ **then**
    Type2Inconsistency = True;
  return [AmtInconsistency, Type1Inconsistency, Type2Inconsistency];

D Inconsistency Detection Algorithm

This section presents a specific inconsistency detection algorithm based on the definition provided in §6.

E Cross-chain Activities Analysis

E.1 Feature Selection

Table 14 lists the 11 features we selected for clustering. Our selection is based on criteria such as interpretability, data characteristics, informativeness, and the value of their distributions. Specifically, we considered the following optional features for each cross-chain transaction graph.

Frequency. Transaction frequency reflects the level of activity and patterns in user behavior. We selected the number of transactions (TxCount) to measure the overall volume of transactions for a user. We chose max weekly frequency (WkFreqMax) to capture the peak weekly transaction frequency and to identify anomalous transaction patterns. We did not select daily frequency due to the short time frame and potential issues with time zones. Given that we only have 12 months of data, using monthly frequency would be too broad, so we opted not to include it.

Table 14: Selected Features of Each Transaction Graph

#	Features	Description
1	TxCount	Total number of transactions.
2	WkFreqMax	Maximum number of transactions in a week.
3	SrcAmtMax	Maximum amount transferred from the source transaction.
4	DstAmtMax	Maximum amount received by the destination transaction.
5	ProfitNum	Number of transactions yielding profit.
6	ProfitSum	Total profit gained across all transactions in a graph.
7	LossSum	Total loss incurred across all transactions in a graph.
8	DegAvg	Average degree of nodes (addresses).
9	DegGini	Gini coefficient measuring the inequality in node degrees.
10	AvgSPL	Average shortest path length between nodes (addresses).
11	Density	Density of each transaction graph, indicating connectedness.

Amount. We aimed to explore extreme cases in the cross-chain activities, so we selected the maximum amount (SrcAmtMax and DstAmtMax) instead of the average amounts. Since both the standard deviation (std) and Gini coefficient for amounts were relatively sparse (with 59% of the data being zero), we did not select these metrics. The values for SrcAmt and DstAmt follow a long-tail distribution. While log transformation is commonly used for such data, we found that clustering results were worse after applying it. Specifically, the silhouette scores for k-means were significantly lower with the same k value. Therefore, we chose not to apply log transformation.

Transfer Cost (Profit / Loss). Transfer profit and loss are key indicators of cross-chain transaction activity. We selected profit transaction count (ProfitNum), profit sum (ProfitSum), and loss sum (LossSum) to measure the user's overall gains or losses. Profit sum represents the total profit across all transactions, while loss sum captures the total loss. Due to the natural fluctuations in profit and loss caused by liquidity variations in cross-chain transaction pools, these fluctuations affect the standard deviation (std) and Gini coefficient, making them unable to accurately reflect the volatility of profit and loss in the transaction graph. Therefore, we did not select them.

Graph. The transaction graph reflects activity among all of a user's addresses. We considered graph features such as *degree*, *average shortest path length*, *average clustering coefficient*, and *density*. Since the average and median degree values were similar, we selected degree average (DegAvg) as the representative feature. We chose the degree Gini coefficient (DegGini) because it better captures the balance of degrees across nodes compared to the degree standard deviation (std). We selected the average shortest path length (AvgSPL) to reflect the overall breadth of the transaction graph. For clustering coefficient and density, since the former captures local density and the latter represents overall density, we focused on global characteristics and selected density.

Diversity. We considered chain diversity (number of unique chains) but did not select it, as 83.6% of transactions involved only two chains, offering low information value. Additionally, we find chain diversity is strongly correlated with transaction count, which we already included as a feature.

E.2 Cluster Methods

Common unsupervised clustering methods include K-means, DBSCAN (Density-Based Spatial Clustering of Applications with Noise), Affinity Propagation, Gaussian Mixture Models (GMM), and Hierarchical Clustering. We evaluated these methods on our dataset and ultimately selected K-means for several reasons.

DBSCAN forms clusters by identifying high-density regions and is effective at finding clusters of arbitrary shapes. However, it tends to classify low-density clusters—such as those representing liquidity pool attacks and arbitrage bots—as outliers, which risks overlooking critical activities. Additionally, in our tests, DBSCAN produced hundreds or even thousands of clusters, requiring extensive manual effort to analyze the results. Other clustering methods face similar problems.

We also experimented with Isolation Forest, an anomaly detection algorithm that isolates observations by randomly selecting features and partitioning the data. Isolation Forest performed well in distinguishing patterns within our dataset; over 95% of ordinary transactions were classified as normal data, while other categories were correctly identified as anomalies. However, Isolation Forest lacks clustering functionality, meaning it cannot group similar anomalous activities together.

In summary, K-means offers effective clustering with computational efficiency by setting an appropriate k. Analyzing cluster centroids and a few samples helps us differentiate between common and anomalous cross-chain transaction patterns, enabling easy detection of unusual activities with minimal effort.

E.3 Elbow Method

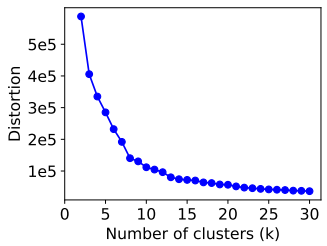


Fig. 11: Elbow Method for Optimal k

The elbow method is used to find the optimal number of clusters by plotting distortion (within-cluster sum of squares) against different k values. The aim is to spot the point where adding more clusters doesn't significantly reduce distortion. As shown in Figure 11, we plotted distortion for k values ranging from 2 to 30. The plot shows a sharp decrease in distortion up to about $k = 7$, after which the decrease slows down, forming an *elbow* between 7 and 10 clusters. This inflection point suggests that increasing k beyond this range gives diminishing returns in reducing distortion.