Journal of King Saud University - Computer and Information Sciences 36 (2024) 101968

HOSTED BY

Contents lists available at ScienceDirect

Journal of King Saud University - Computer and Information Sciences

journal homepage: www.sciencedirect.com

Full length article

A framework for efficient cross-chain token transfers in blockchain networks

Hongyu Guo [a], Haozhe Liang [a], Ju Huang [b], Wei Ou [a,c,*], Wenbao Han [a], Qionglu Zhang [c], Ruizhi Zhang [d]

[a] School of Cyberspace Security (School of Cryptology), Hainan University, Haikou, 570228, Hainan, China
[b] School of Computer Science and Technology, Hainan University, Haikou, 570228, Hainan, China
[c] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
[d] Henan Key Laboratory of Network Cryptography Technology, ZhengZhou, 450001, Henan, China

ARTICLE INFO

Keywords:
Blockchain technology
Token bridge
Cross-chain interoperability
Payment channel
Token standards

ABSTRACT

The proliferation of blockchain technology has resulted in diverse token standards, posing challenges for compatibility, security, and performance in existing cross-chain bridges. This paper introduces a novel framework capable of concurrently facilitating fungible token exchange, as well as the processing of both individual and batch non-fungible tokens (NFTs). We deploy token bridges that meet different token standards to support cross-chain staking and unlocking of ERC20, ERC721, and ERC1155. To minimize both waiting times and handling fees, we relocate processes necessitating frequent transactions and verifications to the sidechain. Additionally, we adopt a batch-processing approach for tokens necessitating cross-chain transfers, leveraging payment channels to facilitate efficiency. The system's reliability is upheld through the validator group. Validators acquire an initial reputation value by making deposits and enhance both their rewards and reputation by successfully completing NFT auction tasks on the sidechain. We use OpenZeppelin's security library functions to standardize token operations, and carefully design the validator's reward, punishment, and reputation mechanisms. Our comprehensive contract security audit and system analysis validate our solution's effectiveness in mitigating common vulnerabilities and internal threats. Implementation and testing with Ethereum and its test network demonstrate substantial reductions in transmission time for key cross-chain token steps by nearly half. Moreover, our framework showcases efficiency and cost-effectiveness with an average gas cost of 693,379.

1. Introduction

Since its inception in 2008 by Nakamoto (2008), Bitcoin has garnered considerable global attention, leading to the widespread recognition of blockchain technology. Thereafter, it has rapidly developed and been applied to various fields. Blockchain technology has been widely adopted, from Internet of Things (IoT) (Reyna et al., 2018; Novo, 2018; Minoli and Occhiogrosso, 2018) and Artificial Intelligence (AI) (Wang et al., 2019; Nguyen et al., 2021) to the metaverse (Gadekallu et al., 2022) and NFT (Non-Fungible Token) (Wang et al., 2021). Major technology companies have invested substantial amounts of money in building the Metaverse in recent years, which is characterized by the fusion between the virtual world and reality. NFTs are arguably the most critical elements in the Metaverse. NFTs were first proposed in Ethereum Improvement Proposals (Entriken et al., 2018) (EIP-721), and they possess unique and irreplaceable characteristics. NFTs can represent anything, including avatars, clothes, pets, game items, and even the land you own in the Metaverse. You can create any object in the Metaverse as an NFT and sell it on the open market, so there are a lot of transactions involved. However, there is isolation between blockchains, especially between heterogeneous blockchains, making it difficult to complete data circulation and value transfer across them. To address this issue, many trading platforms have been established, offering NFT purchase and sales services to users. However, some of these platforms have fallen prey to security breaches. OpenSea,

* Corresponding author at: School of Cyberspace Security (School of Cryptology), Hainan University, Haikou, 570228, Hainan, China.
E-mail address: ouwei@hainanu.edu.cn (W. Ou).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2024.101968
Received 22 September 2023; Received in revised form 19 January 2024; Accepted 11 February 2024
Available online 16 February 2024

a popular NFT trading platform, was hacked on February 19, 2022, resulting in the loss of approximately $1.7 million worth of NFT assets. Similarly, Axie Infinity, a well-known NFT game, announced on March 29, 2022, that it had been hacked, resulting in the loss of 173,600 Ethereum and 25.5 million stablecoin USDC, amounting to a loss of $625 million. On April 25, 2022, BAYC's official Instagram account was hacked, and users were tricked into signing a "safeTransferFrom" transaction, resulting in the transfer of NFTs worth over $10 million to the scammers' wallets. These incidents highlight the security risks associated with entrusting assets to NFT trading platforms. The losses incurred by users due to these attacks are often significant, given the substantial value of cryptocurrencies today.

One of the factors limiting the flow of NFT assets is the immaturity of cross-chain technology. While tokens can typically be transferred within their native blockchain, exchanging or transferring them between different blockchains presents a formidable challenge. Ethereum dominates the current blockchain ecosystem, including the NFT market. Thus, users of other blockchains must first convert their assets into ether through an exchange before participating in transactions. However, this exposes them to additional losses due to exchange rate fluctuations and poses security risks. Consequently, users are eagerly waiting for new cross-chain technologies to enable secure cross-blockchain asset exchange at a low cost.

In recent years, cross-chain technologies have witnessed rapid development, broadly classified into four categories: notary schemes, sidechain/relay mechanisms, hash-locking techniques, and distributed private key control, as outlined by Kannengiesser (2020) (Kannengießer et al., 2020). The notary scheme involves an intermediary that verifies transactions and requires both parties to trust the intermediary. Although this approach is simple and practical, the existence of a third party weakens its security. In sidechains, miners employ Simplified Payment Verification (SPV) to validate transactions on the primary blockchain. Nonetheless, this method is less suitable for multiple blockchains. Hash-locking entails the establishment of a micropayment channel for temporarily securing deposits. While it has found application within the Lightning Network, it remains susceptible to specific vulnerabilities. For instance, if malicious users frequently request transactions and refuse to release the corresponding tokens, the tokens remain locked for an extended period. The distributed private key control scheme enables token exchange through secure multiparty computation and threshold signature technology. However, it is noteworthy that this protocol exhibits substantial complexity and imposes high communication costs. To enable profit-seeking users to showcase their NFTs across multiple blockchain networks and sell them via auctions to maximize returns, new solutions are needed to accomplish this in a safe and efficient manner.

There is limited research on the performance and security aspects associated with cross-chain transfers involving various types and quantities of tokens. Compared with existing blockchain cross-chain solutions, especially those dedicated to accomplishing asset cross-chain tasks (Liu et al., 2021; Hei et al., 2022), our solution can not only facilitates the exchange of equal amounts or exchange rates for a single type of fungible tokens, but also extends its capabilities to handle non-fungible tokens (NFTs). Our proposed solutions cover cross-chain transfers and auction sales.

To address such issues, we propose a protocol that can complete token exchange and transfer in the cross-chain environment. We prioritize a framework that supports the transfer of multiple tokens. This framework incorporates equivalent exchange mechanisms for fungible tokens and employs auction processing for non-fungible tokens, aligning seamlessly with real-world requirements. The main contributions of this manuscript can be summarized as follows:

- Proposed a comprehensive cross-chain transmission framework supporting three token standards: ERC20, ERC721, and ERC1155, which has not been achieved by other previous solutions.
- Optimize cross-chain token auctions by utilizing high-performance sidechains for frequent transactions and confirmations in auction procedures. Propose a validator election method and introduce batch processing of token transfers, effectively reducing the time consumption of related steps by up to half.
- Implement an effective reward and penalty mechanism coupled with a reputation management system for validators in the system. This guarantees incentives for honest validators, making verification opportunities increasingly difficult for malicious validators. Furthermore, introduce a dormancy mechanism to foster fair competition among validators, enhance their enthusiasm within the system, and maintain each validator's reputation and fees at a reasonable level.

The rest of this manuscript is organized as follows: Section 2 introduces the recent research, Section 3 illustrates the proposed scheme and how it works, Sections 4 and 5 demonstrates the experiment outcomes and analysis, Section 6 presents the conclusion of this article.

2. Related works

2.1. Cross-chain asset exchange

As is mentioned in Section 1, there are mainly four cross-chain methods at this stage. The notary scheme is a cross-chain mechanism that relies on weak centralization, similar to real-world intermediary mechanisms. Thomas and Schwartz (2015) proposed an Interledger payment protocol that acts as a notary, allowing users with accounts on multiple blockchains to transact with others. Similarly, Polkadot (Wood, 2016) uses notaries to confirm cross-chain data updates. However, notaries typically have greater authority in this scheme and are responsible for all cross-chain information. They may also penalize users they deem dishonest, rendering this approach risky for public blockchains but suitable for consortium blockchains.

Following the application of the Atomic Swap technique based on hash-locking (Herlihy, 2018) to blockchain, the academic community began designing cross-chain asset exchange protocols based on various methods, including atomic swaps, sidechains, and relays, to facilitate cross-chain asset exchange. Schulte et al. (2019) noted that in order to overcome the limitation of blockchain assets being usable only on the original chain, cross-chain asset exchange protocols must be designed to establish a mechanism for transferring digital assets across different blockchains.

Cross-chain asset exchange can be either centralized or decentralized. Centralized exchanges, such as Coinbase and Binance, have high transaction costs and lack public verifiability in trading activities. This makes them vulnerable to false trading volumes and hacker attacks (Werner et al., 2021). At present, decentralized cross-chain asset exchange is the mainstream research direction of cross-chain asset exchange protocol. BTC-relay (Chow, 2016) is a successful sidechain of Bitcoin Nakamoto (2008) and Ethereum (Wood et al., 2014). It offers a distributed approach to securely verify Bitcoin transactions, enabling relayers to submit transaction headers for verification. This method empowers developers to incorporate new functionalities into other blockchains while preserving the integrity of the existing Bitcoin network. Researchers have additionally introduced proof-of-work sidechains (Kiayias and Zindros, 2019) and proof-of-stake sidechains (Gaži et al., 2019). These sidechains perform basic payment verification of other chains, potentially resulting in a soft fork within the main network.

The XCLAIM protocol proposed by Zamyatin et al. achieves cross-chain asset transactions on existing blockchains by issuing tokens (Zamyatin et al., 2021). It is a versatile framework for achieving untrusted and efficient cross-chain switching in cryptocurrencies. However, a limitation arises in that issuing currency on the blockchain requires contract support for specific functions. Consequently, limited capacity

scripting languages like Bitcoin do not support this operation. Liu Feng et al. introduced the NCASP protocol, which is a cross-chain asset interaction scheme based on improved hashing-lock (Liu et al., 2021). This protocol integrates an account system and smart contract technology to facilitate secure asset exchange between Ethereum and the Fabric alliance chain network. The inclusion of these mechanisms expands the cross-chain capabilities between Fabric and other blockchain platforms. Zakhary et al. proposed AC3WN (Zakhary et al., 2019), a protocol that achieves decentralized atomic cross-chain swaps using a decentralized notary network and smart contracts on the sidechain. This protocol ensures the atomicity and commitment of atomic swaps. However, the introduction of the notary network sidechain increases the complexity of the protocol and reduces its performance. Shadab et al. introduced the 3PP protocol (Shadab et al., 2020), designed to facilitate multi-party cross-chain asset transactions in a general context. This protocol ensures end-to-end property, guaranteeing that payments made by the source parties result in corresponding payments received by the sink parties. The 3PP protocol simplifies multi-party cross-chain asset transactions by converting them into two-party cross-chain transactions and employs a hash-locking mechanism to initiate asset exchanges between the two parties. Lys et al. introduced the RelaySwap protocol (Lys et al., 2020), which advances the concept of relay exchange through a blockchain adaptor, enabling decentralized atomic cross-chain swaps. This protocol eliminates the need for a notary network and sidechains, opting instead to refine atomic swap technology by leveraging the hash-locking mechanism. Tian et al. (2021) implemented a decentralized token transaction protocol that utilizes Ether coin as a transit to enable exchanges between different types of assets. The protocol also enables multiple pairs of users to process cross-chain asset exchanges in parallel based on smart contracts, thereby improving the performance of cross-chain asset exchange. In general, researchers leverage sidechains to enhance the performance of the system in managing cross-chain asset transfers while striving to maintain the decentralized nature of the established system through smart contracts. While most existing cross-chain studies examine the exchange of a single token, our article specifically investigates the cross-chain transfer and exchange of tokens adhering to ERC20, ERC721, and ERC1155 standards.

2.2. Non-fungible token auctions

Researchers have proposed various approaches to enable the circulation of NFTs across different blockchains using smart contracts and cross-chain technologies. Unlike fungible tokens, NFTs typically lack direct exchange equivalence and are predominantly sold through auctions. The inherent characteristics of blockchain, such as openness, transparency, and immutability, make it well-suited for conducting auction processes. Liu et al. (2020) presented a double auction protocol that combines secure multiparty computation and blockchain. This protocol leverages these technologies to enhance fairness and security.

Chen's research focuses on electronic auctions and introduces SBRAC (Chen et al., 2022), a decentralized auction scheme aimed at eliminating the need for trusted third parties. However, these methods are limited to single-chain environments and do not consider cross-chain scenarios. AucSwap (Liu et al., 2021) proposes a cross-chain token exchange protocol that combines atomic swaps (Herlihy, 2018) and the Vickrey auction model (Ausubel et al., 2006). It treats cross-chain token exchange as an auction process and evaluates its effectiveness on the Ethereum platform. Another solution, Practical Agentchain Hei et al. (2022), offers a cross-chain asset exchange system where the agent chain serves as a blockchain-based exchange. It provides users with convenient token exchange services and ensures the usability of exchanged tokens within the ecosystem. Furthermore, despite the establishment of a reward and punishment mechanism for intermediaries in the agentchain, users are not eligible for compensation in cases where malicious intermediaries misappropriate tokens. Additionally, it incorporates data oracles to ensure reliable cross-chain transaction status and fairness in on-chain token exchange. The data oracle (Mohanty, 2018) establishes a connection between smart contracts and external data sources, often utilizing hardware such as SGX (Costan and Devadas, 2016). While the authors claim compatibility with multiple chains, their solution has only been tested on isomorphic Ethereum cross-chains. Unlike previous efforts that merely introduce blockchain technology to eliminate third-party trust intermediaries, our paper introduces a validator collection for hosting auction transactions. This approach addresses the issue of trusted auctioneers more effectively. We have also devised various validator selection methods, incentive and punishment mechanisms, along with reputation value assessment and margin mechanisms. These elements collectively safeguard the interests of all transaction parties and encourage active participation.

2.3. Security and privacy

Since its inception, blockchain has been regarded as an innovative method for storing information, executing transactions, and establishing trust in an open environment. The application of technologies such as hash functions and consensus mechanisms in blockchain is considered a breakthrough in cryptography and network security (Zhang et al., 2019). However, security and privacy issues inherent to blockchain itself are topics of significant concern. Among these, identity authentication closely integrated with distributed systems and approaches like federated learning have become popular research directions.

In 2020, Yazdinejad et al. proposed a decentralized authentication scheme using blockchain to verify distributed patient identities in hospitals, addressing issues of centralization and security in traditional authentication (Yazdinejad et al., 2020). Subsequently, Wang et al. introduced lightweight authentication protocols (Wang et al., 2023, 2022), leveraging concise processing methods like hash functions and XOR operations, effectively serving resource-limited terminal devices such as drones and other mobile devices.

With the continuous improvement of computing resources, researchers have turned to machine learning to address security and privacy challenges in blockchain networks. A blockchain-based federated learning framework was introduced (Lian et al., 2023) for personalized services in healthcare IoT, effectively protecting patients' personal privacy. AP2FL (Yazdinejad et al., 2023b) proposed an auditable privacy-preserving federated learning model using Trusted Execution Environments (TEE), reducing data leakage risks as the client's learning continuously updates distribution and global models.

Deep learning also finds applications in detecting security threats in blockchain. A new model with a non-linear aggregation detection function was proposed (Yazdinejad et al., 2023a), optimizing attack detection based on adaptive fuzzy reasoning. Additionally, considering the characteristics of distributed systems, a framework combining federated learning was constructed (Yazdinejad et al., 2022) to automatically search for network attacks in blockchain-based industrial IoT. Using generative adversarial networks and deep recursive neural networks, another study (Rabieinejad et al., 2023) searched and simulated adversarial attack threats in Ethereum, achieving an accuracy of 82%.

Compared to the aforementioned schemes, our approach strikes a balance between security and efficiency. By leveraging a high-performance sidechain, we significantly enhance the overall performance of the cross-chain exchange. Our gateway supports batch processing of multiple tokens. Furthermore, we have established comprehensive and detailed rules for all participants in the cross-chain auction to safeguard the security of tokens for all parties involved while maintaining decentralization as a fundamental principle. Nevertheless, further endeavors are required for privacy protection.

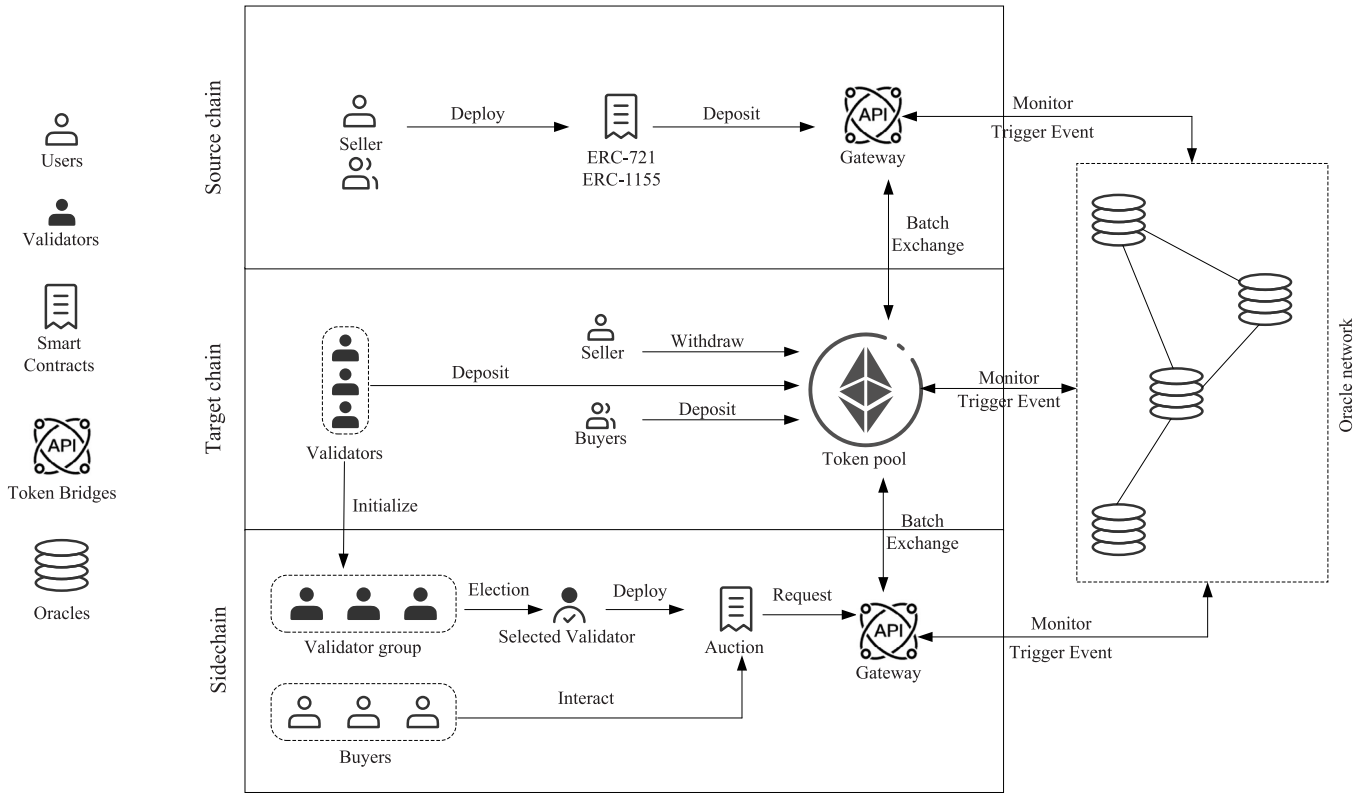

Fig. 1. An overview of the proposed scheme.

3. Proposed scheme

This section outlines the proposed cross-chain token transfer system, depicted in Fig. 1, and subsequently provides an overview of the token transfer process among various components, including gateways, validators, users, and the token pool. The main emphasis of this section centers on ensuring compatibility, fairness, as well as the atomicity and security of token transfers across different blockchains.

3.1. Design goals

- Compatibility: Enable cross-chain transfer of multiple token types and support batch operations.
- Security: Security is an imperative goal, given the involvement of numerous users' token assets. To address this, we propose attracting validators for participation in verification through incentive mechanisms. We evaluate validator behavior using reputation value mechanisms and ensure security by incorporating deposit payments. Malicious validators face penalties, and compromised users are duly compensated.
- Fairness: Ensure that users with a sufficient token balance can participate in the cross-chain transfer process, while validators can compete on a level playing field to offer their services, backed by appropriate reward and penalty mechanisms.
- Efficiency: The proposed scheme should avoid introducing excessively time-consuming frameworks and should instead provide optimizations for the NFT auction process, which often entails frequent transactions and verifications, leading to extended timeframes.

Before presenting the proposed scheme in this paper, it is essential to clarify token standards and the specific functionalities they support, as shown in Table 1. The ERC-20, ERC-721, and ERC-1155 standards are foundational protocols within the Ethereum ecosystem, collectively contributing to the tokenization of digital assets. Their shared application lies in facilitating the representation and exchange of diverse digital assets on the Ethereum blockchain. ERC-20 tokens, being fungible, are commonly employed for applications such as cryptocurrencies or tokenized representations of real-world assets, ensuring seamless interchangeability. In contrast, ERC-721, specializing in non-fungible tokens (NFTs), is instrumental in tokenizing unique assets, including digital art, collectibles, and in-game items, where distinct ownership and individual properties are essential. ERC-1155 serves as a versatile standard accommodating both fungible and non-fungible tokens within a unified framework, streamlining contract deployment and optimizing gas costs. While ERC-20 and ERC-721 cater to specific use cases, ERC-1155 stands out by providing a comprehensive solution for managing diverse token types within a singular contract, offering greater flexibility and efficiency in tokenization applications on the Ethereum blockchain.

To accomplish the aforementioned objectives, we employ gateway components conforming to the ERC1155 standard to streamline cross-chain token batch transfers. Additionally, we establish token gateways that facilitate cross-chain processing for ERC20 and ERC721 tokens. The proposed scheme comprises three blockchains intended to enable secure cross-chain transfers of non-fungible tokens (NFTs) from the source chain to the target chain, along with efficient and equitable token cross-chain exchanges between the sidechain and the target chain. The token transfer and information transmission is carried out through gateways and oracles between the blockchains. The sellers create their NFTs on the source chain and deposit them in the gateway, requesting a cross-chain transfer. Once the gateway confirms the transaction, a deposit event will be triggered. The oracle on the target chain listens to this event and forwards the message to the target chain gateway. Afterward, the gateway on the target chain mints an NFT, using the hash value of the pledged transaction as metadata, and temporarily holds it in the token pool. Once the related transactions are verified, the oracle triggers the auction event. First, the validator group is initialized

Table 1
Token standards and its functions.

Token standards	Fungible	Balance of	Transfer	Approval for transfer	Multiple token	Batch processing
ERC20	✓	✓	✓	✓	×	✓
ERC721	×	✓	✓	✓	×	×
ERC1155	both support	✓	✓	✓	✓	✓

on the sidechain, and then the group obtains representatives through Vickrey-based reverse auction methods. The elected validator deploys the auction contract and gets paid for this time, and must sleep a period of time after the whole process for the interests of other validators and maintain the fairness of the scheme. Buyers on the target chain deposit into the token pool to receive tokens in the sidechain to participate in the auction. After the auction, cross-chain requests from all parties are processed in batches, which not only enhances the efficiency of the scheme but also prevents malicious intermediaries from embezzling users' tokens. The main components and system flow are described below.

3.2. Main component

3.2.1. Gateways

Functionally, a gateway is a smart contract that temporarily holds tokens that need to be transferred. The smart contract should have functions that allow assets to be deposited, withdrawn, and transferred. In this solution, several gateways are implemented to support cross-chain transfer of ERC20, ERC721 and ERC1155 tokens. Any blockchain network that supports smart contract execution environments and adheres to these token standards can utilize these gateways, thereby ensuring excellent compatibility for the proposed solution presented in this paper. The gateway is one of the key components of token transfer, and if not properly configured, it can result in lost or transferred tokens to the wrong account. The contract uses OpenZeppelin's code library to ensure secure and reliable token transfers.

This contract has a dependency on ValidatorManagerContract to enforce validator threshold signatures for withdrawals. The contract defines a map of allowed tokens, which can be used to limit which tokens can be deposited/withdrawn. By default, only tokens that correspond to the specified interface can be received. The contract has a switch to enable/disable the gateway, which is useful for maintenance or security reasons. The contract also uses nonce to prevent replay attacks on withdrawals. After deploying the gateway to the two chains that require token transfers, the ValidatorManagerContract needs to be configured to use a specific validator. Finally, you need to deposit and withdraw tokens to and from the sidechain using the deposit and withdraw functions. Note that withdrawals require a valid threshold signature from the validator group.

In these contracts, two main functions are provided: withdraw and deposit. The deposit function is used to deposit tokens into the gateway. This function interacts with the receiving token interface of the gateway, prompting the gateway to generate a deposit certificate. Users can subsequently utilize this certificate to withdraw tokens from the destination chain gateway. The withdraw function is used to withdraw tokens from the contract and transfer them to the caller's address. For ERC1155 gateways, they can handle token transfers in batches. Both functions emit events to notify the caller of the transaction and the oracles. The algorithm 1 below describe more details of these functions.

Transferring tokens between blockchains involves multiple steps and parties, which increases the risk of things going wrong. Some common areas where errors can occur include:

- Invalid transaction parameters: If the transaction parameters are not properly set, such as incorrect gas limits or insufficient fees, the transaction may not be successfully executed.
- Smart contract bugs: If there are bugs in the smart contract code used for the transfer, it can lead to unexpected behavior, loss of tokens, or even a complete failure of the transfer.

Algorithm 1 onERC1155BatchReceived

function ONERC1155BATCHRECEIVED
  $\{\_from, \_to, \_ids[], \_amounts[], data\}$
  $message : createMessageWithdraw$
  $Withdrawtoken, keccak256(amount, contractAddress))$
  $vmc.checkThreshold(message, \_signersIndexes)$
  $nonces[msg.sender]++$
  $bal \leftarrow balanceOf(IERC1155(contractAddress))$
  if $bal < amount$ **then**
    $IERC1155GatewayMintable(contractAddress)$
    $safeMint(address(this), amounts[], ids[])$
  end if
  $IERC1155(contractAddress).safeTransfer$
  $emitCross-chain$
end function

- Network congestion: If the network is congested with many transactions, it may take longer for the transfer to be confirmed, or the transaction may fail altogether.
- Insufficient confirmation: If the transaction is not properly confirmed by the destination blockchain, the tokens may not be successfully transferred.

To address such issues, the withdraw function takes as input the amount of tokens to be withdrawn, the contract address of the token, and a set of signatures from validators. It first creates a message using the $createMessageWithdraw$ function, which includes information about the withdrawal request and the caller. It then verifies the signatures using the $vmc.checkThreshold$ function, which checks that the required number of validators have signed the message. The $\_signersIndexes$ is an array of integers representing the indexes of the validators who have signed the message. It is used to ensure that the required number of validators have signed the message before the withdrawal can be completed. If the signatures are valid, the function increments the nonce of the caller, checks the balance of the contract for the token, and mints more tokens if necessary. Finally, it transfers the requested amount of tokens to the caller's address and emits a TokenWithdrawn event.

The deposit function takes as input the amount of tokens to be deposited and the contract address of the token. It simply transfers the tokens from the caller to the contract, emits an TokenReceived event.

We have implemented ERC20 gateway and ERC721 gateway to facilitate cross-chain transfers of these tokens. The ERC721 gateway contract inherits from the ERC20 gateway contract, and in addition to calling the original basic functions, it also allows for the deposit and withdrawal of ERC721 tokens. It is important to note that these functions require specific parameters such as token ID, contract address, and type for proper execution. We have also added the latest ERC1155 gateway to support bulk transfer of tokens.

3.2.2. Validators

Validators are Ethereum contracts responsible for validating transactions and blocks. The contract offers functions to modify the validator group, such as $rotateValidators$, which updates the validators and their respective powers; setValidator, which updates the address of a specific validator; and setQuorum, which changes the minimum number of validators required to reach consensus. There is a contract that manages

a list of validators and their reputations on a blockchain. The initial set of validators can only be created once during the deployment of the contract and can only be modified later using the interface provided by the contract. The function *rotateValidators* was responsible for updating the list of validators and their corresponding reputations in the contract state. It first checked that both arrays had the same length and that they were greater than zero. Then, it calculated the total reputation of all validators provided. Finally, it updated the contract state with the new list of validators and their reputations, and emitted an event (named *ValidatorSetChanged*) to notify interested parties of the change.

Token transfers through gateways require multiple validators to validate them. We have a built-in threshold for agreeing transactions in the system, and the gateway will check whether the number of signatures of the transaction reaches the threshold to decide whether to transfer tokens. To prevent malicious users from taking over the validator set, the contract uses a threshold-based approach to validate changes to the set. This means that a certain proportion of the total voting reputations of the validators must sign off on a proposed change before it can be accepted. The constructor takes *num* and *denom* variables as arguments to set the required threshold. The *checkThreshold* function takes as input a message hash and arrays of signer indexes, v, r, and s values. Algorithm 2 describe its details. The function checks if the number of signatures provided is less than or equal to the number of validators in the contract and also checks that the arrays have the same length and are greater than zero. It then calculates the hash of the message using the keccak256 function and verifies the signatures for each provided index. If the signature is malleable, the function skips it; otherwise, it uses the ecrecover function to recover the signer's address, which is then verified against the list of validators. Finally, the function checks if the voted power is greater than or equal to the required threshold. The SafeMath library is used throughout the contract to perform secure arithmetic operations and prevent integer overflow and underflow vulnerabilities.

Security is an imperative goal, given the involvement of numerous users' token assets. To address this, we propose attracting validators for participation in verification through incentive mechanisms. We evaluate validator behavior using reputation value mechanisms and ensure security by incorporating deposit payments. Malicious validators face penalties, and compromised users are duly compensated.

Algorithm 2 checkThreshold

```
function checkThreshold(_message, _signersIndexes)
    sig_length ← length of _v
    require(sig_length ≤ validators.length)
    require(sig_length == _signersIndexes.length)
    hash ← keccak256(_message)
    votedPower ← 0
    Skipmalleablesignatures
    continue
    signer ← ecrecover(hash, _v[i], _r[i], _s[i])
    require(signer = validators[_signersIndexes[i]])
    votedPower ← votedPower + powers[_signersIndexes[i]]
    require(votedPower * threshold_denom ≥ totalPower * threshold_num)
end function
```

3.2.3. Oracles

Oracle enables initially isolated smart contracts to access data beyond the blockchain, playing a crucial role in connecting smart contracts with the real world. Combined with gateways, cross-chain transfers can be quickly responded to and less verification is demanded. Chainlink is a decentralized oracle network that aims to provide secure and reliable access to off-chain data for blockchain-based applications. The Chainlink network is composed of a decentralized network of nodes, called "oracles", that source data from multiple data providers and deliver it on-chain in a tamper-proof manner. One of the main advantages of decentralized oracles, such as Chainlink, is their ability to provide trustless and verifiable data to smart contracts. By using multiple oracles to source data and using a consensus mechanism to ensure data accuracy, decentralized oracles can provide reliable data without a single point of failure or reliance on a single data provider. In addition, it supports many networks such as Ethereum, Loom, Fabric, and so on. Therefore, we can use Chainlink as the data source for our scheme.

Employing external oracles guarantees the independence of the blockchain network, safeguarding its security and decentralization. This approach also enhances the accuracy and reliability of oracle-supplied data, as it is not inherently associated with any particular blockchain network. Furthermore, it can contribute to system scalability and performance improvements by outsourcing computational and storage demands to an external service.

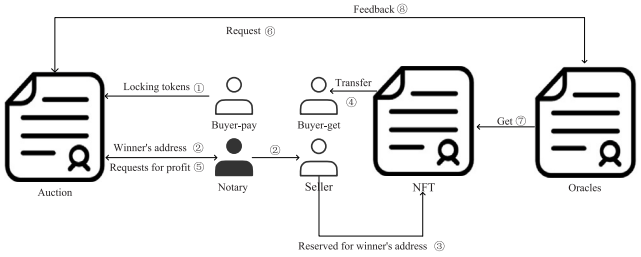

Fig. 2. Process of the cross-chain auction.

3.3. System process

Based on our scheme, a complete process of NFT cross-chain auction is:

1. Network configuration. Gateways, validators and oracles initialized.
2. Validator election. Compared to multi-signature authentication, we choose to select one candidate from the validator set at a time to complete the transaction as a notary.
3. NFT auction. Users bid to buy NFT, and tokens are correctly transferred between addresses on the blockchain according to the user's request and the transaction implementation process.

During the initialization phase, the system undergoes the configuration of cross-chain gateways and oracles within each blockchain network. Subsequently, users expressing a willingness to partake in the verification of cross-chain auction transactions join the validator set by pledging a deposit to the gateway. Within this set, members collaboratively oversee transaction verification, as well as the entry and exit of fellow members, facilitated through autonomous voting. Validators successfully completing the verification process stand to receive tokens and reputation rewards. Instances of transaction failure resulting from malicious behavior, such as delays, lead to the corresponding deduction of tokens and reputation.

Consider a scenario where a user on Chain A aims to sell a non-fungible token (NFT) and acquire token assets on Chain B. In this case, the user must initially pledge tokens on Chain A to the gateway. Upon confirmation of the relevant transaction, a certificate is generated on Chain B, attesting to the ownership of the NFT by user A. The validator on Chain B functions as an auction house, with its behavior constrained by a reward and punishment mechanism, as well as a deposit, effectively mitigating third-party risks. Users from Chains A and B have the liberty to participate freely in the auction, exchanging ERC20 tokens through the gateway for auction participation. In the event of a successful auction, the winning user gains ownership of the NFT asset, while remaining users can opt to retain tokens for subsequent auctions or exchange original assets at the gateway.

Fig. 2 mainly describes the token transfer during the auction, and more details about the scheme will be discussed afterwards.

3.3.1. Network configuration

Before establishing a connection to the oracle, the gateway and the related token contracts should be deployed correctly on the blockchain. After initializing the validator group and reputations, the notary elected for the verification set must deposit a certain amount in the gateway to protect the rights and interests of users. This deposit will also have a certain impact on the notary's verification weight.

3.3.2. Validator election

The most convenient way to achieve cross-chain asset transfers is through the use of notaries as the main component in the cross-chain method. However, the main risk associated with this method is the possibility of malicious notaries dumping intermediate assets or intentionally failing to verify cross-chain transactions. To address these issues, we have adopted a reverse auction scheme based on the Vickrey auction model, which selects the most suitable notary for each auction.

To prevent malicious notaries from participating in cross-chain transactions and to prevent potential notary collusion, we have improved upon the Vickrey auction model by incorporating the reverse auction method to simulate the notary election process. This scheme takes into account various factors such as the deposit amount of the notary account, historical reputation, service fees, and more to select the optimal notary for cross-chain transaction users. Once a legal auction transaction request is received, the administrator account notifies the notary group to elect a notary to coordinate and complete the asset cross-chain transfer. During the notary election process, each notary account submits its own quotation, and the administrator account selects the appropriate notary based on the reverse auction method. The administrator collects each notary's historical fee, deposit amount, credibility, and other factors to select the notary with the lowest service fee for the user while ensuring security requirements are met. This helps to ensure that only trustworthy notaries are selected for cross-chain transactions, preventing any fraudulent or malicious activity.

Fairness is a goal extending to all participants. Protection of cross-chain users' interests is paramount, but we also strive to ensure that honest validators have opportunities for profit. To prevent validators with excessively high reputation values from monopolizing the verification market, we mandate a resting period for validators successfully completing cross-chain transactions. This ensures other validators have equal opportunities to compete and profit.

The management of notaries' offers involves a gaming process. Excessively high offers can reduce user participation in trading, as it places a financial burden on them, while excessively low offers may fail to attract enough honest notaries, increasing the risk of malicious notaries. The reverse auction mechanism is particularly suitable for scenarios where multiple service providers compete to offer services to users. While this is more beneficial to buyers, the interests of notaries are not compromised, as rational notaries offer their services for profit. However, to create a secure cross-chain auction system, every effort must be made to prevent malicious notaries from participating in the transaction. Hence, the reverse auction mechanism alone cannot fully satisfy the requirements. To avoid notaries exaggerating the true cost or maliciously competing with low prices, we introduce the Vickrey auction model, combined with the reverse auction mechanism. During the election process, each validator submits a bid in secret without knowing the offers of others, and the contract collects and calculates all bids. The detailed process is described in algorithm 3. It retrieves the validator's quotes and finds the second lowest price. After computing the reputation average on the validator group, it will find the first validator as the notary that meets all the demands.

To balance the interests of both parties, we investigated the average fees of the last 1000 transactions on the Ethereum mainnet. The average transaction fee was found to be 0.0012 ETH, with 62.8% of transactions not transferring any amount. It is important to note that the transaction fee is solely based on the gas consumed by the contract and the gas base fee, and not on the transfer amount. Our research revealed that fees typically account for less than 10% of the transfer amount.

Algorithm 3 Validator Election

```
function ELECTION( )→ address
    totalPower ← 0
    minFee ← fees[0]
    secondMinFee ← fees[1]
    minFeeIndex ← 0
    secondMinFeeIndex ← 1
    for i ← 0 to validators.length do
        totalPower ← totalPower + powers[i]
        if fees[i] < minFee then
            secondMinFee ← minFee
            secondMinFeeIndex ← minFeeIndex
            minFee ← fees[i]
            minFeeIndex ← i
        else if fees[i] < secondMinFee then
            secondMinFee ← fees[i]
            secondMinFeeIndex ← i
        end if
    end for
    powerAverage ← totalPower ÷ validators.length
    while powers[secondMinFeeIndex] < powerAverage and
  secondMinFeeIndex < validators.length − 1 do
        minFee ← secondMinFee
        minFeeIndex ← secondMinFeeIndex
        secondMinFee ← fees[++secondMinFeeIndex]
    end while
    if powers[secondMinFeeIndex] ≥ powerAverage then
        return validators[secondMinFeeIndex]
    else
        return address(0)
    end if
end function
```

To reduce the burden on users, we recommend that notaries charge a service fee of 5% of the value of the NFT being auctioned, while not intervening in the bidding process. Assuming that the final bid for the NFT is $finalPrice$, and the notary charges a service fee of $txFee$, the amount the buyer needs to pay would be calculated as follows:

$$payment = finalPrice + txFee + gasPrice$$

After successfully processing a transaction and earning a fee, the validator must enter a period of inactivity. This is to prevent monopolization and provide opportunities for other validators to participate in the auction process. Overall, this approach can help ensure fair and reasonable fees for users, while also enabling notaries to provide their services for profit.

3.3.3. NFT auction

The sidechain is a crucial component for implementing NFT cross-chain auctions, supporting services such as user registration, asset mortgage, asset transactions, and oracles through smart contracts. The purpose of establishing the sidechain is twofold:

1. NFTs are typically minted on the Ethereum mainnet, and the auction process for NFTs requires frequent transactions from buyers. By constructing a sidechain anchored to the mainnet, related transactions can be rapidly verified on the sidechain. This not only speeds up the NFT auction process but also saves some gas fees for auction participants.
2. Token transfers between blockchains can also be facilitated more easily through the notary group. While the notary mechanism has been touted as the most efficient way to exchange assets,

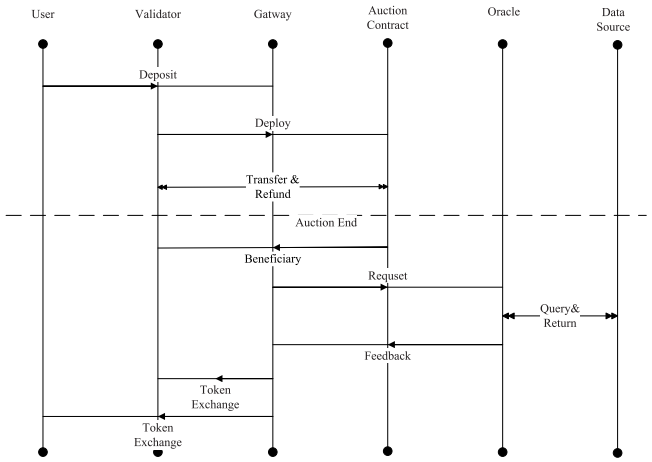

Fig. 3. Flow of tokens between entities.

its centralization risk has been criticized to some extent. Our solution uses deposit and election mechanisms to ensure proper functioning while effectively controlling risk.

The notary elected by the users will deploy the auction contract. Once deployed, buyers can bid, request a refund, and access auction information through the contract's invocation method. To ensure buyer protection, the contract includes a refund invocation method. Regardless of whether the auction is in progress or has ended, buyers can request a refund, allowing them to participate in the transaction with greater freedom and without the need to trust the notary. For the seller, the contract ensures that the current highest bid is irrevocable, ensuring that a buyer who meets the requirements will always be available to complete the transaction. To increase transaction efficiency, a quote filtering strategy has been implemented, where transactions below the current highest quote are not confirmed. If a buyer's bid is rejected, they will receive an alert to place a higher bid. However, blindly increasing one's bid is not recommended. Therefore, the contract provides several query methods, including checking the current highest bid, enabling buyers to make informed decisions and increase the transaction success rate. Fig. 3 illustrates the specific steps.

When the auction is completed, we shift our focus to the cross-chain requirements of different tokens, specifically, batch cross-chain transactions. In the scenario presented in this article, the auction transaction necessitates frequent communication between the buyer and the contract. If these interactions are considered as cross-chain requests, the time required to complete the auction would be significantly prolonged. On the other hand, centralized processing of token transfers prior to and following the auction dramatically reduces the time needed to process cross-chain requests.

Efficiency is another pursuit, achieved through measures such as leveraging high-performance side chains and implementing the batch cross-chain transfer method of tokens proposed in this article. These measures effectively reduce communication consumption.

The buyer and seller complete the final token transfer on the target chain. In the previous steps, the system has confirmed that the auction is complete and the NFT are allowed to be transferred to the buyer's address. Apart from that, the buyer's funds, the notary's profits, and the seller's profits are temporarily stored in the token pool. It is actually a gateway on the target chain, but it temporarily stores the tokens of many users, so it is easier to describe its function at this time by using the token pool for the time being. Normal notary programs rely entirely on notaries and run the risk of lost funds and uncoordinated transfers. In this paper, we adopt the following approach to complete the token transfer on the relay chain and target chain through the notary and administrator accounts in collaboration.

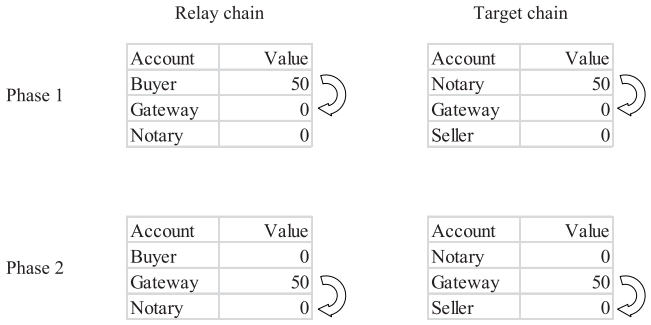

Fig. 4. Cross-chain transfers in two-phase mode.

The transfer process of tokens is shown in Fig. 4. The entire process can be divided into two stages. The act of depositing tokens into the gateway is considered token locking. During this stage, when the buyer needs to make a payment to the seller, a challenge arises because their tokens reside on different chains. To facilitate the trade, the involvement of a notary is necessary to assist in the completion of the transaction. Nevertheless, it is important to acknowledge that utilizing a notary for fund transfers always carries inherent risks. Additionally, cross-chain transfers are sensitive to the network environment, making it challenging to guarantee atomicity, further adding to the complexity and potential uncertainties involved.

To address these challenges, our proposed solution involves entrusting the gateway with the management of the tokens belonging to the involved parties. This transaction entails two transfers: the initial transfer from the buyer on the sidechain to the notary, and the subsequent transfer from the notary on the target chain to the seller. The gateway acts as an intermediary by receiving the tokens from both transfers. Subsequently, it provides a withdrawal proof, allowing the parties to actively withdraw the tokens locked within the gateway based on the provided proof. It is important to note that this process requires verification by the oracles, ensuring that only valid withdrawal proofs are accepted and utilized. We have implemented an enable and disable status for the deposit and withdrawal interfaces of the gateway. In the initial stage, only user deposits are accepted, while withdrawals are not permitted. Once the gateway receives a deposit, it sends a deposit certificate to the oracle machine. The withdrawal interface is opened by the oracle machine once a sufficient number of deposit certificates have been received. When a user submits a withdrawal request, the gateway checks the current status and rejects the request if withdrawals are not allowed at that time. The gateway is managed by contract logic and cannot refuse to pay or lock tokens maliciously. The existence of the gateway removes the need for users to trust the notary, because the notary's funds are not withdrawn until the second phase is completed.

4. System analysis

In this section, we will first analyze the potential areas of errors in the cross-chain transfer and auction process, and then demonstrate how our scheme overcomes these risks.

4.1. Security concerns

During a cross-chain transfer, several points that may cause errors include:

- Incorrect or insufficient deposit: The sender may deposit an insufficient amount of tokens or the wrong type of tokens, resulting in a failed transfer.
- Network congestion: The transfer may fail due to network congestion or other network-related issues.

- Gateway failure: The gateway may fail to relay the transfer request or handle the transfer process properly, resulting in a failed transfer.
- Oracles: The oracles responsible for relaying information between the source and target chains may fail to function properly or may be delayed, resulting in a failed transfer.
- Contract errors: The smart contract responsible for managing the transfer may contain errors or bugs that prevent the transfer from completing successfully.
- Intermediary Failure: In cases where a third party is engaged in cross-chain transfers, the possibility of improper execution or malicious actions by the intermediary can lead to transfer failures.
- Insufficient gas: If the gas limit provided for the transaction is too low in Ethereum-based blockchains, the transfer may not complete successfully.
- Scalability: Scalability stands as a crucial metric in assessing the efficacy of the system. In the absence of support for a broad spectrum of blockchains or the ability to seamlessly integrate with emerging technologies, sustained long-term development becomes challenging.

4.2. Correctness of the methods

In this section, we demonstrate how our scheme mitigates the aforementioned risks through testing. We utilized the contract audit tool Mythril to conduct a thorough analysis of the contracts, and we are pleased to report that the analysis was completed successfully without any detected issues or vulnerabilities. We verify the accuracy of each component's functions using truffle-based testing framework. For the gateway component, the tests include preventing signature withdrawals from being reused, prohibiting re-entry, and filtering out invalid signatures. We also conducted tests for various scenarios and edge cases related to depositing and withdrawing all kinds of tokens. These scenarios include ensuring that the withdrawal amount does not exceed the user's balance, ensuring that the user's equity is above the threshold for withdrawal, and preventing signature withdrawals from being reused. For edge cases, when a user tries to deposit a token that does not meet the expected type, the gateway will reject it and throw a badToken event. These tests ensure that insufficient deposit will be alerted and contract errors will be avoided. As for gateway failure, the validator group have the ability to temporarily disable or enable gateway so that users will not use bad gateways. And for validators, they can only be added or removed by themselves and rotate their positions to serve users.

Assuming that User A deposits 10 Token A from Chain A into the gateway with the intention of exchanging them for 20 Token B on Chain B, several factors influencing cross-chain transactions become concerns. Blockchain confirmation delays, oracle communication delays, and potential malicious activities by third parties stand out among these factors. Primarily, if network communication delays impede the confirmation speed of the blockchain, A's tokens will remain in the gateway without disappearing. The impact of oracle delays is analogous. Although A's tokens are temporarily locked, cross-chain transactions are not perpetually ongoing. In case of a transaction timeout, users have the option to reclaim their tokens. Regarding potential malicious activities, the validator set is designed to effectively mitigate malicious behavior. Even if a malicious node successfully disrupts the cross-chain transaction, the collateral deposited by the malicious node serves as compensation for the user's tokens. Therefore, users can confidently engage in transactions, assured that safeguards are in place to address potential disruptions.

The oracle in our system is a decentralized oracle network, which uses multiple independent nodes to fetch, verify, and transmit data to ensure accuracy and reliability. It bolsters the reliability of smart contracts by offering secure and immutable inputs and outputs.

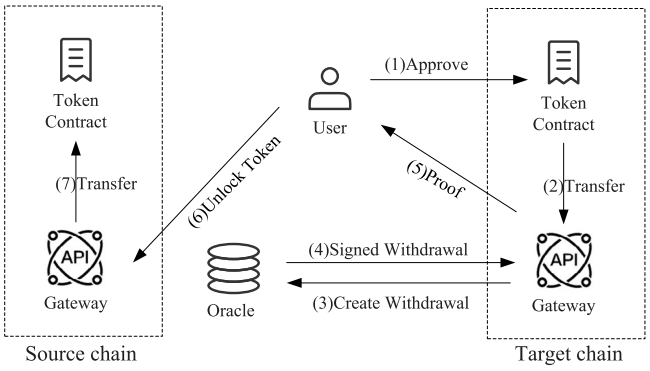

Fig. 5. Work flow of token withdrawals.

In the final stage of the token transfer, we introduced an intermediary through elections to facilitate the completion of the transfer. However, this approach is not completely centralized as staking tokens restrict the intermediary's behavior, who would aim to make a profit. Even if a malicious intermediary tries to disrupt the cross-chain transaction, the gateway and validator can verify the transaction and reimburse the intermediary's pledged tokens to the user. Hence, This method maintains the properties of decentralization and trustlessness.

4.3. Threat model

In our threat model, we categorize potential malicious attackers into two groups: those motivated by profit and those aiming to disrupt the smooth execution of cross-chain transactions. The first type of attacker is likely to attempt token withdrawal by forging gateway withdrawal proofs or exploiting reentrancy vulnerabilities in the contract. They may also impersonate regular users in an attempt to repeatedly withdraw tokens that do not belong to them. The second type of attackers may engage in activities such as spreading disruptive information throughout the network, intentionally monopolizing port resources, or refusing to fulfill their designated responsibilities.

4.4. Against external attacks

In this section, we will analyze and present strategies to defend against the attack models discussed in the previous section.

Against profit-seeking users. A potential vulnerability in the two-way cross-chain transfer of tokens arises during the return of tokens from the target chain to the source chain. Users may exploit this step by repeatedly withdrawing tokens. In our scheme, Fig. 5 illustrates the process of a user requesting to transfer tokens back. After the user authorizes the token transfer from the target chain to the gateway, the gateway generates a withdrawal request. Subsequently, this request undergoes verification, is signed by the oracle, and transmitted back to the user. Armed with the proof, the user can then unlock the original tokens within the mainnet gateway. Importantly, the user cannot forge a valid proof of withdrawal as they do not have access to the oracle node's private key. Moreover, the presence of a nonce in the proof prevents its reuse by users. Furthermore, to mitigate risks effectively, the contract always sets the token balance to 0 before proceeding with token unlocking. This precautionary measure prevents the contract from being vulnerable to reentrancy attacks.

Against malicious attackers. When it comes to attackers intentionally disrupting or delaying the cross-chain transfer of tokens, the most accessible identity they can assume within the system is that of a user. However, users have limited influence over the system's process, and their primary course of action would involve initiating numerous requests to the gateway. This would result in a substantial consumption of transaction fees for the attacker, making the attack costly. Moreover,

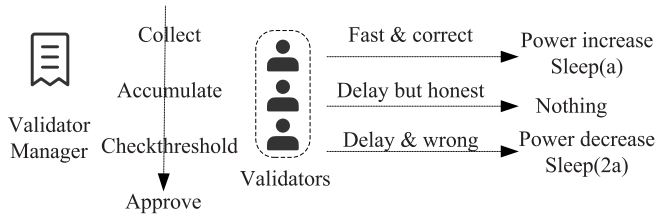

Fig. 6. Consequences of different behaviors of validators.

the blockchain operates on a distributed network where the verification of transactions does not rely on a single node alone. Therefore, even if some nodes fail, other nodes can still perform transaction verification, rendering such attempts ineffective.

Should the attacker manage to obtain the identity of a validator, they would gain involvement in two crucial processes within the system: verifying and signing cross-chain transfer requests, as well as competing for NFT auction qualifications. In the former process, it is highly improbable for an attacker to forge an obviously false proof due to the severe penalties associated with such actions. Instead, the attacker's most likely behavior would involve intentionally not responding, aiming to deliberately delay the verification process of the cross-chain transfer. During the multi-validation process, the validator management contract accumulates powers as it collects signatures from validators. Assuming there are ten validators in total, with an initial multi-signature threshold set at q/p, and the total weight of the validator set denoted as n, the contract can approve the transaction once the current weight reaches or exceeds nq/p. In the scenario where other validators consistently submit their verification results honestly and in a timely manner, for an attacker to impact the final result, their weight must surpass $(p-q)n/p$. However, this situation is inherently implausible as the system adjusts the weights of each validator based on their behavior. This adjustment mechanism prevents the emergence of super nodes with excessive influence over the outcome, ensuring a more balanced and fair result (see Fig. 6).

As depicted in the figure above, a fast and reliable validator will have their power increased after each transaction. However, to prevent the concentration of excessive power and the monopolization of the verification process, such validators will also be required to enter a sleep period. On the other hand, honest validators will not face penalties for submitting their proofs late. However, if their signature results differ from the majority, their weight will be reduced, and they will be required to undergo a longer sleep period to avoid interfering with the verification process. In fact, when a validator's weight becomes too low, other validators can collectively vote on whether to remove them from the validator group. Therefore, it becomes challenging for a single malicious attacker to impede the normal operation of the system.

Regarding the process of competing for NFT auction qualifications, attackers may attempt to attract users by offering lower service fees and then either not responding or embezzling tokens. However, our scheme employs the reverse Vickrey auction method, which is a second-price auction model combined with a reverse auction. As a result, the validator with the second lowest service fee will ultimately qualify. If an attacker offers a significantly lower service fee than the overall median, their reputation value and power will be reduced. Without an accomplice, it becomes challenging for the attacker to accurately guess the second lowest price. Therefore, we believe that our scheme effectively mitigates the aforementioned attack model.

4.5. Scalability

From a technical standpoint, the solution presented in this article relies on the collaboration of components such as gateways and oracles to facilitate the cross-chain transfer of tokens. Therefore, any blockchain network that supports a Turing-complete smart contract language, along with a corresponding smart contract virtual machine and execution environment, can theoretically integrate seamlessly with the approach outlined in this article. Currently, there exist numerous blockchain networks that endorse the Ethereum Virtual Machine (EVM), establishing EVM as a standard for many blockchain platforms. Consequently, the scalability of the proposed solution in this article is expected to be robust.

Regarding the supported token types, the solution accommodates cross-chain operations compliant with three token standards: ERC20, ERC721, and ERC1155. Notably, these token standards enjoy widespread popularity, with ERC20 being particularly prominent. As of the present date (2024.1.16), Ethereum hosts a total of 1275 token contracts adhering to the ERC20 standard, indicating strong compatibility for the token aspect of the proposed solution in this article. As long as subsequent tokens adhere to these established standards, the solution is poised to offer continued support.

In terms of the performance limitations of the solution outlined in this article and its responsiveness to increasing transaction volumes, a detailed exploration will be undertaken in the subsequent chapter.

5. Performance evaluation

In this section, we present a performance analysis of the system, which includes details of the experimental implementation, test results, comparisons and analysis.

5.1. Experiment setup

We utilized Georli, an Ethereum test network, as the source chain, and a local ganache ethereum network as the target chain for token transfer. We built a sidechain of the target chain in conjunction with Loom SDK, a tool that provides a second-layer expansion solution for Ethereum. Loom employs DPOS as a consensus mechanism to offer fast and affordable authentication services, making it ideal for processing numerous transactions. Additionally, Chainlink provides off-chain data services for smart contracts in the system.

The experimental hardware environment consisted of a computer with an 11th Gen Intel(R) Core(TM) i7-11800H @ 2.30 GHz CPU, 32 GB DDR4 RAM, and a Ubuntu20.04 virtual machine with 8 cores, 8G RAM, and 100G SSD. We implemented components such as gateway and validator by combining open source tools Truffle and Loom SDK and defined the process of notary election and auction. To verify the functionality of each contract and method, we initially benchmarked them. Subsequently, we performed several comparative experiments to validate the performance optimization achieved by our proposed method. Finally, we compared the experimental results with those obtained from other approaches.

5.2. Evaluation results

We first tested the gas and time consumed by deploying related components in different networks, and we also counted the average confirmation time and gas consumption of the last 1000 blocks of the Ethereum mainnet. The gasFee of the local Ethereum is set to be the same as the current Ethereum mainnet (21GWei) to be as close to the real situation as possible. We have extracted transaction data from approximately 500 blocks and will now present the following graph based on the collected data.

The results of Table 2 show that our scheme can consume less gas cost under the same gas base fee. Simultaneously, we have deployed multiple standard tokens and their corresponding adapted gateways. Upon analysis, we observe that the deployment and transaction costs for the three tokens are relatively similar. However, it is worth noting that the deployment costs of their gateways gradually increase. This is primarily due to the complexity associated with implementing

Table 2
Gas used of deploying the contracts.

Smart contracts	Ganache	Goerli
Validator Manager	1,604,826	1,982,844
ERC20 Gateway	951,839	1,578,454
ERC721 Gateway	2,412,686	3,721,975
ERC1155 Gateway	3,707,302	3,708,388
ERC20 Token	1,006,727	1,563,218
ERC721 Token	1,493,530	2,112,679
ERC1155 Token	2,679,498	2,885,875

Table 3
Time cost of deploying the contracts.

Smart contracts	Ganache	Goerli
Validator Manager	1.51	10
ERC20 Gateway	1.82	12
ERC721 Gateway	2.83	12
ERC1155 Gateway	3.67	12
ERC20 Token	1.67	11
ERC721 Token	1.81	12
ERC1155 Token	1.92	12
Total(s)	15.23	81

Table 4
Gas cost of the contracts from Practical AgentChain.

Smart contract	Gas cost
Agent Contract	1,878,008
Token Contract	607,490

Table 5
Time and gas cost of the algorithms from Practical AgentChain.

Algorithms	Time cost(s)	Gas cost
askforToken	21	149,930
askforWithdraw	22	100,617
proof	11	62,293
judge	17	76 447
arbitration	18	58,409

an ERC1155 gateway to support the handling of multiple types and quantities of tokens.

In addition, Table 3 reflects the significant difference in confirmation time between the Ethereum testnet and the sidechain, which indicates that it is necessary for us to place the auction process that requires frequent transactions in the sidechain. The performance of our smart contract on both the layer-1 and layer-2 networks of Ethereum relies on the block generation speed of the respective networks, and optimization methods are limited. To ensure efficient operation of our scheme, we opt to place additional transactions on an autonomous and controllable sidechain. This approach allows us to seek improved efficiency in the execution of our solution.

We conducted a comparative analysis of the gas and time costs associated with our contracts in relation to similar works (see Tables 4 and 5).

The test results mentioned above were obtained from the practical agentchain Hei et al. (2022), which utilizes agent contracts to manage accounts on two blockchains and facilitate user token swaps. Upon reviewing the table, it is evident that our solution's contract requires slightly more gas compared to theirs, although it does not exceed double the amount. We believe the primary reason for this disparity is that their solution only supports the circulation of a single homogeneous token, whereas our gateway enables cross-chain pledging and unlocking of multiple token standards such as ERC20, 721, and 1155. Consequently, our solution incurs additional gas costs. Furthermore, based on the test results obtained from the Goerli network, our contract's interaction method demonstrates a comparable time requirement to theirs. With the exception of the proof algorithm, our approach exhibits a lower time consumption than theirs. However, when compared

Table 6
Time and gas cost of the algorithms.

Algorithms	Time cost(ms)	Gas cost
safeMint	103	233,733
safeTransferfrom	156	196,749
onERC1155Received	139	188,284
onERC1155BatchReceived	275	412,201
check threshold	1030	1,006,384
validator election	393	525,588

Table 7
Stress test result for the auction contract.

Send Rate	Max latency	Min latency	Average latency
976.3	18.03	10.35	12.23

to transactions verified in the sidechain, there remains a substantial performance gap. Therefore, our objective is to migrate time-consuming steps to the sidechain wherever feasible.

In contrast to peer works, the gas consumption of our management components has decreased by 11%. Thanks to the side chain environment, the deployment time of related components and the time consumption of key algorithms have been significantly reduced, approximately one-tenth of the original values.

However, it is noteworthy that the gas consumption of the three token gateways in this article is approximately 25% higher than that of its peers. This discrepancy arises because our approach supports three composite Ethereum token standards for both fungible and non-fungible tokens, while peers exclusively considered a single fungible token issued on a custom blockchain.

We are using the truffle framework to write test scripts for the main components in the system, mainly involving access to the gateway, as well as the multi-signature verification and election process of the verifier. The results are shown in the Table 6 below. The test results indicate that the gas consumption and confirmation time for the contract method calls are relatively low. However, the most time-consuming process is the multi-signature verification of the validator set for gateway transactions, specifically in the $checkthreshold$ function.

We also employed the script to conduct a stress test on the auction contract by submitting 10,000 bids to the contract from 10 different accounts within a 10-second period. The resulting test outcome is as follows:

It can be seen from Table 7 that the send rate in the actual test is 976.3, with a total of 10,000 bid transactions received. Thanks to the excellent performance of the sidechain, the auction contract maintains an average response time of 12.23 ms, with a maximum response time of 18.03 ms. While the actual auction process may take longer due to waiting for user quotations, the contract's ability to maintain high throughput and fast response time is crucial for providing optimal user experience. Fig. 9 depicts the time consumed by an ERC1155 token during the process of cross-chain transfer and auction. Based on our tests, the token generation time on the source chain is approximately 12 s, and the pledge time within the gateway also takes around 12 s. We have set a waiting time of 60 s for the auction process to collect and process user quotations. Finally, new tokens are generated on the target chain, which takes approximately 0.7 s in terms of time consumption. We conducted a total of five rounds of testing, and Fig. 7 displays the specific gas consumption for each round. Overall, a round of testing encompasses approximately 100 blocks, and each round yields similar results. In each round, there are several peaks in gas consumption, primarily due to the multiple verifications of the validator set required for cross-chain request validation. Apart from these peaks, the gas consumption remains relatively stable throughout the testing process.

Among the recorded data from Fig. 8, the contract gas consumption ranges from a minimum value of 0 to a maximum value of 2,563,410, with a median value of 335,423. A total of 504 blocks were analyzed,

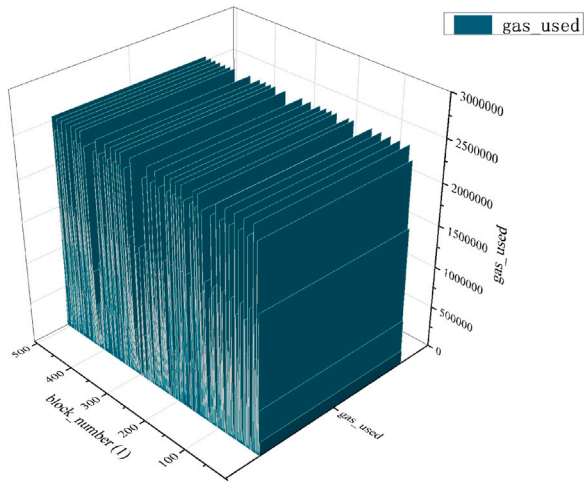

Fig. 7. Overall gas cost of the cross-chain process.

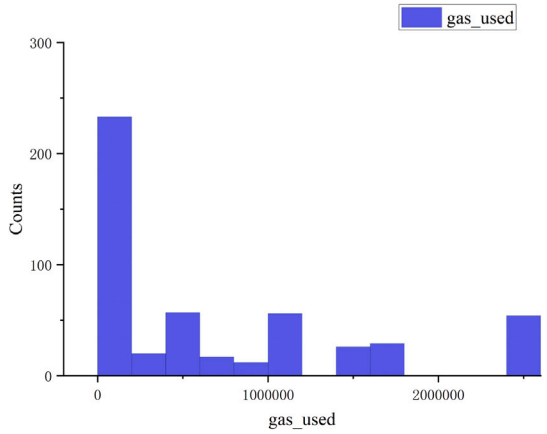

Fig. 8. Mathematical statistics of gas consumption.

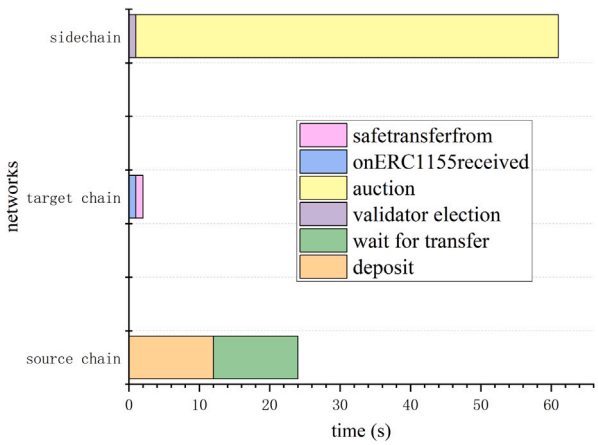

Fig. 9. Time cost of the cross-chain process.

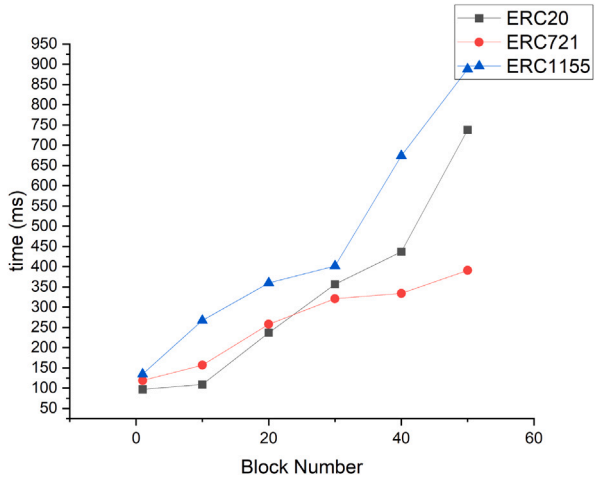

Fig. 10. Time cost of the different tokens.

and the average gas consumption is calculated to be 693,379.5. Both the average and median values indicate that the gas consumed by the contracts within the system and their associated calling methods is considered acceptable. Based on the statistical analysis depicted in Fig. 8, it is evident that out of the total blocks examined, 339 blocks consume less than one million gas, and 450 blocks consume less than two million gas. This implies that approximately 67.2% of the blocks exhibit a remarkably low level of gas consumption. Notably, there are 233 blocks with gas consumption hovering around 100,000, which strongly indicates the cost-effectiveness of our contract implementation.

As observed in Fig. 9, the most time-consuming aspect is the waiting time for the NFT auction, which is currently set to 60s by the system. This duration can be dynamically adjusted as needed. However, based on the stress test results presented in Table 7, the processing delay of the sidechain does not significantly impact the cross-chain process in this context. Nonetheless, it is important for the contract to accept as many quotes as possible to maximize the benefits derived from the auction process. If the auction process occurs on either the source chain or the target chain, it is anticipated that both the time required and the associated transaction fees would significantly rise.

The scalability of the system is also one of the important indicators of performance. First, we tested the gateway's processing time for various tokens as the number of transactions sent to the gateway increased.

As was shown in Fig. 10, as the number of transactions continues to increase, the time required by the gateway generally shows an upward trend. From an early stage, the processing time required for ERC20, 721 and 1155 increases in sequence. As the number of transaction requests increases, ERC20 and ERC1155 require additional verification of the number of tokens and corresponding token types due to batch processing, resulting in a larger increase in overhead, and since ERC721 only supports a single process, the total time consumed does not change much.

We used the truffle framework to write test examples for the gateway to test the cross-chain request capabilities that the gateway component can receive and its processing capabilities. We progressively augment the number of users in the test case, with each user initiating cross-chain requests at a fixed frequency of 5 times per second. As depicted in Fig. 11, the oracle machine exhibits robust performance, effortlessly handling this user load with effective request forwarding. In contrast, the gateway experiences a gradual decline in processing time when confronted with 6 to 8 users frequently dispatching cross-chain transaction requests. Further increases in the user count result in elevated network latency and gas consumption, constrained by the limitations of our hardware infrastructure.

In fact, the total number of cross-chain transactions that occur in each period does not change significantly. Even if the verification speed of the side chain is faster, it is always necessary to wait for the verification results of the main chain to complete the cross-chain transaction. Therefore, the performance bottleneck here does not drag down the actual cross-chain transaction speed.

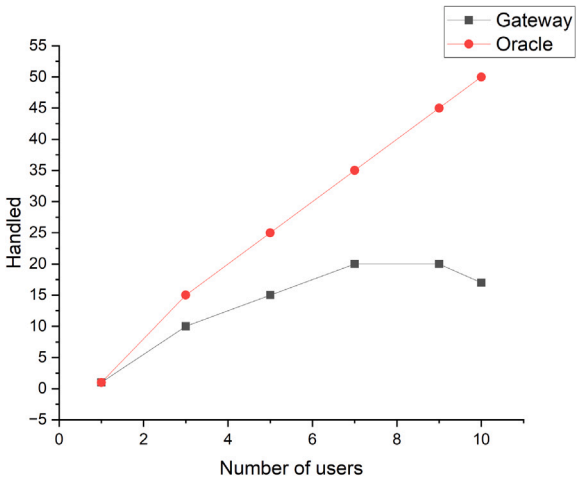

Fig. 11. Changes in the performance of system components with growing users.

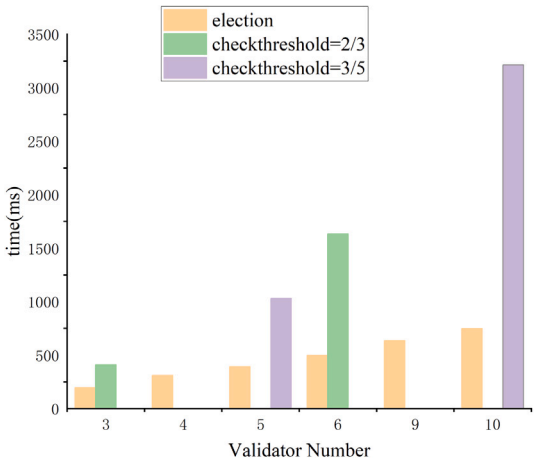

Fig. 12. Time cost of election and multi-signature.

Although our local testing between the VM and the host assumes an ideal communication latency, it is important to note that in real-world scenarios, the latency is constrained by physical performance limitations. In this case, we have set the polling time of the oracle to 1s to account for these realistic limitations. Throughout the entire cross-chain transfer process, it is important to note that, except for the auction contract, which requires a sufficient amount of time to proceed, the time consumption of the remaining parts is considered acceptable.

We conducted observations on the validator set to validate the efficacy of our proposed election and dormancy mechanisms in preserving fair competition. We initialized 10 validators, each starting with an initial reputation score of 100. Without a dormancy mechanism, it becomes evident that, given the same service price, validators with high reputation will consistently be elected. This could lead to the undesired scenario where validators, leveraging their accumulated high reputation, undertake auction tasks at elevated service prices. In reality, our dormancy mechanism, in conjunction with the reward and punishment mechanism, collaborates to uphold the credibility of validators and maintain service prices at a reasonable level.

5.3. Performance comparison

In this solution, the two most time-consuming processes are the multi-signature verification of validators and the cross-chain transfer of tokens. We propose optimizing these processes through validator election and batch cross-chain methods. The specific implementation and experimental results are presented below.

The confirmation time of validators depends on the amount of validators and the threshold set by default. To ensure security, we assume that the threshold must be greater than $1/2$. However, to balance performance and security requirements, we have decided to set the threshold to $3/5$. This minimizes the requirements for validators while still meeting the necessary security standards. Assuming that the number of verifiers is 5n, the minimum number of signatures that the gateway needs to verify is 3n. For each signature, the gateway needs to recover its public key according to r, s, v message, and then traverse the validator list of the $ValidatorManager$ contract to verify whether the signature is valid. The time complexity of this operation is $O(n^2)$. In contrast, the time complexity of validator election is $O(n)$, as it iterates through all validators to compute the total power and identify the two validators with the lowest fees. The while loop responsible for adjusting the minimum fee iterates at most n-2 times. Consequently, the overall time complexity amounts to $O(n)$.

We set the thresholds of 2/3 and 3/5 to test the performance of the validators and compare it with the validator election approach. The detailed experimental results are shown in Fig. 12. These results demonstrate that the time needed for the validator election method gradually increases as the number of validators grows. Conversely, the time spent on the multi-signature verification process sharply escalates with both the increasing number of validators and the verification threshold. These findings demonstrate that the validator election method is more efficient than the multi-signature verification process, especially as the number of verifiers and the verification threshold increase. Hence, the proposed validator election method can notably enhance the performance of the gateway access verification process in cross-chain transactions.

This approach represents a trade-off between performance and absolute security, as observed during our local experiments. We adopt a tolerance towards the presence of malicious validators, implementing punitive measures for any detected malicious behaviors. Furthermore, we streamline the multi-factor authentication process by electing representatives. In actual blockchain networks, despite the presence of substantial computing power for the verification process, the block generation speed of the overall blockchain often falls short of meeting production requirements. However, we remain confident that our approach is effective and viable.

As was illustrated in Section 3, token transfers are completed in two stages after the auction. In essence, the two stages of the process involve depositing into the gateway followed by withdrawal from it, the specific process is presented in Fig. 4. For cross-chain token exchanges that require the participation of multiple parties, we split the process into two batches for centralized processing. This divides unordered access operations into two types of repeated transactions, making it easier for validators and gateways to centralize similar transactions within a short time period. Moreover, the two-step process also helps ensure the security of funds for both parties involved in the transaction. After completing the first stage, the intermediary only holds bridge tokens, while their own mortgaged tokens remain locked. If they refuse to execute the second stage, their pledged tokens will be unable to be withdrawn from the gateway. This ensures that the intermediary has a stake in completing the transaction and helps to prevent malicious behaviors.

To validate our approach, we employed different methods to process cross-chain transactions and tested the gateways of both the target chain and the sidechain. Specifically, we randomly sent transactions such as deposits and withdrawals to the gateway using different accounts and measured the time required for the final transfer of tokens. The experiment aimed to assess the efficiency and effectiveness of our approach in facilitating secure and seamless cross-chain exchanges. The results of the experiment are presented in Fig. 13, which provides valuable insights into the performance of the proposed approach.

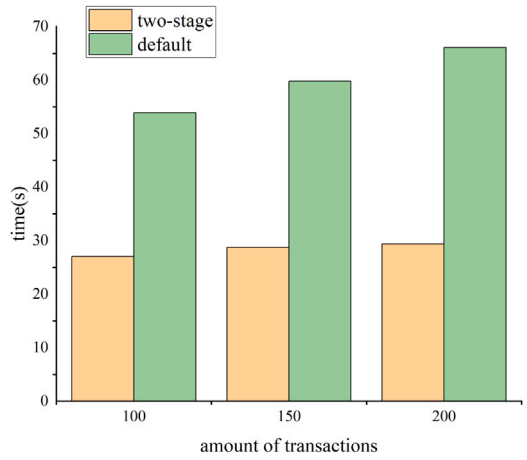

Fig. 13. Time cost of batch mode and default.

We configured the oracle's polling interval to 1 s and established a threshold of 15 s to determine when to change the processing direction of cross-chain requests. Within a defined window period, the target chain's gateway exclusively receives deposits and forwards the deposit proofs to the sidechain. Conversely, during this period, the sidechain gateway only allows users to mint tokens based on the received proofs, without accepting any deposits. In the subsequent window period, the roles and functions of the two gateways are exchanged, enabling the seamless fulfillment of users' one-way cross-chain requirements. From the figure, it is evident that the two-stage batch processing method for cross-chain token transfers is significantly more time-efficient compared to real-time processing of individual cross-chain requests from users. The processing time of both methods generally increases linearly with the number of transactions, but the growth rate of the two-stage processing method is slower. This indicates that the two-stage batch processing approach is more efficient and can handle a larger volume of transactions without experiencing a substantial increase in processing time. The batch processing method ensures that within 200 transactions, the processing time does not exceed two windows. In an ideal scenario where user request volume aligns perfectly with the gateway's processing window, all user transfer requests can be processed within a single window. However, in practice, the timing and direction of user-initiated cross-chain requests are uncertain, requiring individual requests to wait until the next window for processing. Comparatively, the real-time response method, under the same transaction volume, requires approximately three to four window periods to process all requests. This highlights the effectiveness of the batch processing method as an efficient expansion approach for handling cross-chain requests.

Honest users play a crucial role in facilitating the successful cross-chain transfer of tokens, while malicious users can potentially disrupt our optimization method. This is because the oracle relies on a sufficient number of withdrawal proofs to enable the withdrawal interface. To mitigate this, our initial approach is to establish a waiting threshold to ensure that the time required to complete the cross-chain transfer does not exceed two waiting thresholds. Users who do not withdraw within the allocated time can wait for the next window to initiate their withdrawal, as long as they possess a valid withdrawal proof.

6. Conclusions

This paper presents compatible framework for facilitating complex cross-chain token transfers of multiple types and quantities, and propose practical approaches to improve efficiency. By integrating the latest gateway components compliant with the IEEE international cross-chain standards, we propose an optimization framework that caters to both fungible and non-fungible tokens. The framework's core concept revolves around off-chain processing of transactions that demand high time and transaction costs. It leverages expansion techniques like sidechains and payment channels to strike a balance between performance and security in token cross-chain transfers. To ensure decentralization and the security of user token assets, we introduce a validator election method in conjunction with the reverse Vickrey auction model. Furthermore, we batch process one-way token transfers through off-chain channels. Our scheme-based analysis and experimental results demonstrate that our framework significantly reduces gas consumption levels and confirmation latency for various token cross-chain transfers.

Nevertheless, our work does have certain limitations. The validators which are responsible for processing cross-chain transactions, introduce complex parameters such as multiple verification thresholds, reputation, and dormancy mechanism. Achieving optimal performance in token cross-chain transfers requires effectively handling and balancing these parameters. Additionally, we also need to incorporate more advanced methods suitable for distributed systems, such as federated learning. Despite the inherent anonymity of blockchain identities, the association of transaction behavior with real-world data can potentially expose and analyze the identity behind a specific address. Consequently, we intend to bolster the privacy and security safeguards of this solution through the implementation of federated learning. Moreover, we aim to integrate cross-chain standard components to explore the best practices for performance, security, and scalability of cross-chain token transfers in heterogeneous blockchains.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the Joint Funds of National Natural Science Foundation of China (Grant No. U23A20304), Hainan Provincial Natural Science Foundation of China (621RC508), State Key Laboratory of Information Security [2022-MS-04], Henan Key Laboratory of Network Cryptography Technology (Grant/Award Number: LNCT2021-A16), the Science Project of Hainan University (KYQD(ZR)-21075).

References

Ausubel, L.M., Milgrom, P., et al., 2006. The lovely but lonely vickrey auction. Comb. Auctions 17, 22–26.
Chen, B., Li, X., Xiang, T., Wang, P., 2022. SBRAC: Blockchain-based sealed-bid auction with bidding price privacy and public verifiability. J. Inf. Secur. Appl. 65, 103082.
Chow, J., 2016. Btc relay.
Costan, V., Devadas, S., 2016. Intel SGX explained. Cryptology ePrint Archive.
Entriken, W., Shirley, D., Evans, J., Sachs, N., 2018. Eip-721: Erc-721 non-fungible token standard. Ethereum Improv. Propos. (721).
Gadekallu, T.R., Huynh-The, T., Wang, W., Yenduri, G., Ranaweera, P., Pham, Q.-V., da Costa, D.B., Liyanage, M., 2022. Blockchain for the metaverse: A review. arXiv preprint arXiv:2203.09738.
Gaži, P., Kiayias, A., Zindros, D., 2019. Proof-of-stake sidechains. In: 2019 IEEE Symposium on Security and Privacy. SP, IEEE, pp. 139–156.
Hei, Y., Li, D., Zhang, C., Liu, J., Liu, Y., Wu, Q., 2022. Practical AgentChain: A compatible cross-chain exchange system. Future Gener. Comput. Syst. 130, 207–218.
Herlihy, M., 2018. Atomic cross-chain swaps. In: Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing. pp. 245–254.
Kannengießer, N., Pfister, M., Greulich, M., Lins, S., Sunyaev, A., 2020. Bridges between islands: Cross-chain technology for distributed ledger technology.
Kiayias, A., Zindros, D., 2019. Proof-of-work sidechains. In: International Conference on Financial Cryptography and Data Security. Springer, pp. 21–34.
Lian, Z., Wang, W., Han, Z., Su, C., 2023. Blockchain-based personalized federated learning for internet of medical things. IEEE Trans. Sustain. Comput.
Liu, L., Du, M., Ma, X., 2020. Blockchain-based fair and secure electronic double auction protocol. IEEE Intell. Syst. 35 (3), 31–40.

Liu, W., Wu, H., Meng, T., Wang, R., Wang, Y., Xu, C.-Z., 2021. AucSwap: A vickrey auction modeled decentralized cross-blockchain asset transfer protocol. J. Syst. Archit. 117, 102102.

Lys, L., Micoulet, A., Potop-Butucaru, M., 2020. Atomic cross chain swaps via relays and adapters. In: Proceedings of the 3rd Workshop on Cryptocurrencies and Blockchains for Distributed Systems. pp. 59–64.

Minoli, D., Occhiogrosso, B., 2018. Blockchain mechanisms for IoT security. Internet Things 1, 1–13.

Mohanty, D., 2018. Advanced programming in oraclize and IPFS, and best practices. In: Ethereum for Architects and Developers. Springer, pp. 151–179.

Nakamoto, S., 2008. Bitcoin: A peer-to-peer electronic cash system. Decentralized Bus. Rev. 21260.

Nguyen, D.C., Ding, M., Pathirana, P.N., Seneviratne, A., 2021. Blockchain and AI-based solutions to combat coronavirus (COVID-19)-like epidemics: A survey. IEEE Access 9, 95730–95753.

Novo, O., 2018. Blockchain meets IoT: An architecture for scalable access management in IoT. IEEE Internet Things J. 5 (2), 1184–1195.

Rabieinejad, E., Yazdinejad, A., Parizi, R.M., Dehghantanha, A., 2023. Generative adversarial networks for cyber threat hunting in ethereum blockchain. In: Distributed Ledger Technologies: Research and Practice. ACM New York, NY.

Reyna, A., Martín, C., Chen, J., Soler, E., Díaz, M., 2018. On blockchain and its integration with IoT. Challenges and opportunities. Future Gener. Comput. Syst. 88, 173–190.

Schulte, S., Sigwart, M., Frauenthaler, P., Borkowski, M., 2019. Towards blockchain interoperability. In: International Conference on Business Process Management. Springer, pp. 3–10.

Shadab, N., Houshmand, F., Lesani, M., 2020. Cross-chain transactions. In: 2020 IEEE International Conference on Blockchain and Cryptocurrency. ICBC, IEEE, pp. 1–9.

Thomas, S., Schwartz, E., 2015. A protocol for interledger payments. URL https://interledger.org/interledger.pdf.

Tian, H., Xue, K., Luo, X., Li, S., Xu, J., Liu, J., Zhao, J., Wei, D.S., 2021. Enabling cross-chain transactions: A decentralized cryptocurrency exchange protocol. IEEE Trans. Inf. Forensics Secur. 16, 3928–3941.

Wang, K., Dong, J., Wang, Y., Yin, H., 2019. Securing data with blockchain and AI. IEEE Access 7, 77981–77989.

Wang, W., Han, Z., Gadekallu, T.R., Raza, S., Tanveer, J., Su, C., 2023. Lightweight blockchain-enhanced mutual authentication protocol for UAVs. IEEE Internet Things J..

Wang, Q., Li, R., Wang, Q., Chen, S., 2021. Non-fungible token (NFT): Overview, evaluation, opportunities and challenges. arXiv preprint arXiv:2105.07447.

Wang, W., Yang, Y., Yin, Z., Dev, K., Zhou, X., Li, X., Qureshi, N.M.F., Su, C., 2022. BSIF: Blockchain-based secure, interactive, and fair mobile crowdsensing. IEEE J. Sel. Areas Commun. 40 (12), 3452–3469.

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

Wood, G., 2016. Polkadot: Vision for a heterogeneous multi-chain framework. White Paper 21, 2327–4662.

Wood, G., et al., 2014. Ethereum: A secure decentralised generalised transaction ledger. Ethereum Proj. Yellow Pap. 151 (2014), 1–32.

Yazdinejad, A., Dehghantanha, A., Parizi, R.M., Hammoudeh, M., Karimipour, H., Srivastava, G., 2022. Block hunter: Federated learning for cyber threat hunting in blockchain-based iiot networks. IEEE Trans. Ind. Inform. 18 (11), 8356–8366.

Yazdinejad, A., Dehghantanha, A., Parizi, R.M., Srivastava, G., Karimipour, H., 2023a. Secure intelligent fuzzy blockchain framework: Effective threat detection in iot networks. Comput. Ind. 144, 103801.

Yazdinejad, A., Dehghantanha, A., Srivastava, G., 2023b. Ap2fl: auditable privacy-preserving federated learning framework for electronics in healthcare. IEEE Trans. Consum. Electron..

Yazdinejad, A., Srivastava, G., Parizi, R.M., Dehghantanha, A., Choo, K.-K.R., Aledhari, M., 2020. Decentralized authentication of distributed patients in hospital networks using blockchain. IEEE J. Biomed. Health Inform. 24 (8), 2146–2156.

Zakhary, V., Agrawal, D., Abbadi, A.E., 2019. Atomic commitment across blockchains. arXiv preprint arXiv:1905.02847.

Zamyatin, A., Al-Bassam, M., Zindros, D., Kokoris-Kogias, E., Moreno-Sanchez, P., Kiayias, A., Knottenbelt, W.J., 2021. Sok: Communication across distributed ledgers. In: International Conference on Financial Cryptography and Data Security. Springer, pp. 3–36.

Zhang, R., Xue, R., Liu, L., 2019. Security and privacy on blockchain. ACM Comput. Surv. 52 (3), 1–34.