國立臺灣大學電機資訊學院資訊網路與多媒體研究所

碩士論文

Graduate Institute of Networking and Multimedia

College of Electrical Engineering and Computer Science

National Taiwan University

Master's Thesis

基於區塊鏈與零知識證明之隱私與公平性保護售票務系統

A Privacy- and Fairness-Preserving Ticketing System Using Blockchain and Zero-Knowledge Proofs

符嘉文

Foo, Jia Wen

指導教授: 洪一平 博士

Advisor: Yi-Ping Hung, Ph.D.

中華民國 115 年 1 月

January 2026

doi:10.6342/NTU202600625

國立臺灣大學碩士學位論文
口試委員會審定書
MASTER'S THESIS ACCEPTANCE CERTIFICATE
NATIONAL TAIWAN UNIVERSITY

基於區塊鏈與零知識證明之隱私與公平性保護票務系統

A Privacy- and Fairness-Preserving Ticketing System Using Blockchain and Zero-Knowledge Proofs

本論文係 符嘉文 （學號 R10944057）在國立臺灣大學資訊網路與多媒體研究所完成之碩士學位論文，於民國 115 年 1 月 29 日承下列考試委員審查通過及口試及格，特此證明。

The undersigned, appointed by the Graduate Institute of Networking and Multimedia on 29 January 2026 have examined a Master's Thesis entitled above presented by FOO, JIA-WEN (student ID: R10944057) candidate and hereby certify that it is worthy of acceptance.

口試委員 Oral examination committee:

（指導教授 Advisor）		

系（所）主管 Director:

doi:10.6342/NTU202600625

誌謝

若說大學時期仍能恣意馳騁，研究所的我則可謂一路匍匐前行。回首這段歷程，實在難以置信自己能夠堅持到今天，而這一切都離不開身邊眾人的支持與幫助。

首先，謹向指導教授洪一平老師以及各位口試委員致上誠摯的感謝，感謝老師們在口試過程中提出寶貴的建議，指出論文中尚待改進之處，使本研究得以更加完善。若於口試招待或答辯過程中有任何不周之處，尚祈老師們海涵。

同時，也特別感謝洪老師提供在威如實習的機會，成為我就學期間重要的經濟來源。實習期間所累積的智能合約、零知識證明與網站技術相關知識與實務經驗，對本研究具有關鍵性的幫助。在此一併感謝王皓正學長、徐誌鴻學長、陳煬升學長、謝喬恩學姐，以及其他一同共事的同仁，能與各位合作學習，令我獲益良多。

亦感謝實驗室同儕們營造出溫暖且互相扶持的學習環境，能在研究所期間與各位相遇實屬幸運。謝謝大家在學業與研究上的協助，尤其感謝區塊鏈小夥伴陳冠廷同學與曾宏鈞同學的陪伴與討論。

一路走來，親友們的關心與鼓勵始終是我重要的支撐，特別感謝胞妹符嘉尹，在我面對畢業相關事務時給予許多實際的幫助與陪伴。最後，我想對自己說聲謝謝，縱使曾無數次停下腳步，卻始終沒有選擇放棄。

doi:10.6342/NTU202600625

摘要

線上售票系統中，尤其在熱門活動的情境下，黃牛炒票的手法層出不窮，嚴重影響票券交易的公平性。現有解決方案多仰賴實名制以限制重複購票，然而此方法需要使用者揭露大量個人資訊，並使售票平台成為高度敏感資料的集中管理者，增加隱私外洩與濫用的風險。本研究旨在提出一套兼顧使用者隱私保護與交易公平性的線上售票系統，避免在防制黃牛的同時犧牲使用者隱私。

本研究提出名為 BlozkTix 的線上售票系統。系統建於以太鏈上，透過智慧合約執行購票與票券狀態管理，以確保交易流程的透明性、公正性與抗攻擊能力。同時，系統引入真人唯一性（Proof of Personhood, PoP）證明作為身分驗證機制，在使用者無需揭露個人資訊的情況下仍能有效限制重複購票行為。

在虛擬貨幣逐漸普及、交易紀錄公開可驗的背景下，如何在公開帳本中避免交易紀錄與現實身分關聯已成為一大挑戰。為此，本研究採用多重匿名身分設計，將不同生命週期的身分證明分離，讓使用者在購票、選位等不同階段的行為完全無法被關聯分析。

此外，透過採用裝置通行密鑰（passkey）作為票券身分的衍生來源，使匿名身分可透過裝置驗證即時重建，而不需以可轉移的私鑰或助記詞形式儲存，降低票券身分被轉移、外流與遺失的風險。

實作與評估結果顯示，本研究提出之系統在合理的計算時間與交易成本下，

doi:10.6342/NTU202600625

有效實現跨購票生命週期的身分不可連結性，同時維持票券交易的公平性與實用性，提供可行的隱私保護型線上售票解決方案。

關鍵字：售票系統、零知識證明、隱私保護、區塊鏈、去中心化應用

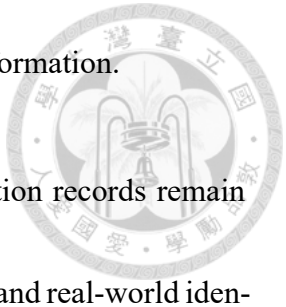
doi:10.6342/NTU202600625

Abstract

In online ticketing systems, especially for high-demand events, ticket scalping has become increasingly prevalent and severely affected fairness in ticket distribution. Existing solutions often rely on real-name registration to restrict repeated purchases. However, such approaches require users to disclose extensive personal information and place ticketing platforms in the role of centralized custodians of highly sensitive data, increasing the risk of privacy leakage and misuse. This thesis aims to design an online ticketing system that preserves user privacy while maintaining transaction fairness, avoiding the trade-off between anti-scalping measures and privacy protection.

This work proposes BlozkTix, an online ticketing system built on the Ethereum blockchain. Smart contracts are used to perform ticket purchases and manage ticket states, ensuring transparency, fairness, and resistance to tampering. To replace traditional real-name verification, the system adopts Proof of Personhood (PoP) as the identity verification mechanism, allowing users to prove their uniqueness and eligibility. Hence, the system

doi:10.6342/NTU202600625



effectively limiting repeated purchases without revealing personal information.

As cryptocurrencies become increasingly prevalent and transaction records remain publicly verifiable, preventing the linkage between on-chain activities and real-world identities becomes a critical challenge. To address this issue, BlozkTix introduces a lifecycle-scoped anonymous identity design, in which identity proofs used at different stages of the ticket lifecycle are explicitly separated. As a result, user actions during ticket purchase, seat selection, and other stages cannot be correlated by any party.

Furthermore, the system derives ticket-related anonymous identities from device-bound passkeys. By allowing identities to be re-derived through device authentication rather than stored as transferable private keys or mnemonic phrases, this reduces the risk of identity transfer, leakage, or loss, at the same time improving usability for end users.

Implementation and performance evaluation results show that the proposed system achieves cross-stage unlinkability of user identities under acceptable computational cost. At the same time, it preserves fairness and practicality in ticket transactions, providing the practical solution of a privacy-preserving online ticketing system.

Keywords: Ticketing System, Zero-knowledge proofs, Privacy-preserving, Blockchain, Decentralized Application

doi:10.6342/NTU202600625

Contents

doi:10.6342/NTU202600625

doi:10.6342/NTU202600625

doi:10.6342/NTU202600625

doi:10.6342/NTU202600625

List of Figures

doi:10.6342/NTU202600625

List of Tables

doi:10.6342/NTU202600625

Chapter 1 Introduction

Ticketing systems play a critical role in large-scale events such as concerts, sports games, and exhibitions. As demand for popular events continues to grow, traditional online ticketing platforms increasingly struggle with issues of fairness, security, and user trust. In particular, automated scalping, unfair ticket allocation, privacy-invasive identity verification, and opaque resale mechanisms have become persistent problems. These challenges highlight the need for ticketing systems that can ensure fair access while respecting user privacy.

One of the most significant challenges in modern ticketing systems is ticket scalping. Scalpers exploit automated scripts, multiple accounts, or coordinated bot networks to purchase large quantities of tickets within seconds of release, depriving genuine users of fair access. Prior studies and industry reports have shown that automated bots can acquire a substantial portion of tickets for high-demand events within seconds, leading to systematic exclusion of genuine users and persistent price inflation in secondary markets[10, 31, 32]. Although platforms have introduced countermeasures such as rate limiting, CAPTCHA, and queue mechanisms, these approaches remain imperfect and are frequently circumvented by sophisticated attackers.

To mitigate scalping, many ticketing platforms have adopted real-name registration

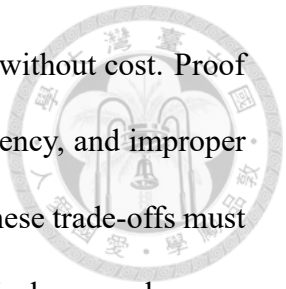
doi:10.6342/NTU202600625

systems that bind tickets to users' legal identities. While real-name systems can increase accountability and limit mass purchases, they introduce serious privacy concerns. Users are required to disclose sensitive personal information that is unrelated to the ticketing process itself, violating the principle of data minimization. Furthermore, centralized storage of identity data creates single points of failure, exposing users to risks of data breaches, misuse, and long-term surveillance. These concerns raise fundamental questions about whether fairness and accountability can be achieved without sacrificing user privacy.

Recent advances in blockchain technology and zero-knowledge cryptography offer promising alternatives to traditional centralized ticketing architectures. Blockchain provides a transparent and tamper-resistant execution environment, enabling verifiable enforcement of ticketing rules without reliance on a trusted intermediary. Zero-knowledge proofs further allow users to demonstrate properties such as uniqueness, eligibility, or ownership without revealing their underlying identities. Together, these technologies suggest the possibility of designing ticketing systems that are both fair and privacy-preserving.

However, existing blockchain-based ticketing systems also exhibit important limitations. Many proposed designs record ticket ownership, transfers, or seat assignments directly on-chain, making user behavior publicly observable and linkable[12, 42]. While such transparency improves auditability, it can also expose sensitive information such as attendance patterns and social relationships[12]. Moreover, blockchain-based systems cannot inherently prevent off-chain transactions, as users may sell private keys or credentials outside the protocol. In practice, wallet-based identity management often relies on mnemonic phrases or key custody mechanisms that are unfamiliar to general users, introducing usability and security challenges that hinder real-world adoption.

doi:10.6342/NTU202600625



Despite their advantages, zero-knowledge proof systems are not without cost. Proof generation and verification introduce computational overhead and latency, and improper protocol design may lead to unintended privacy leakage or misuse. These trade-offs must be carefully considered when applying zero-knowledge techniques in large-scale, user-facing systems such as ticketing platforms[2, 4].

In this thesis, we present BlozkTix, a privacy- and fairness-preserving online ticketing system that leverages blockchain and zero-knowledge proofs to address the limitations of existing approaches. Rather than binding tickets to persistent user identities, the proposed system adopts a lifecycle-scoped anonymous identity model, which allows eligibility to be verified at each stage such as ticket purchase and seat registration, while preventing the system from linking user actions across stages. In addition, Proof of Personhood is employed to restrict repeated purchases without requiring real-name registration, and entrance verification can be performed offline without repeated identity checks. Finally, the system integrates passkey-based authentication to derive ticket-related identities, reducing reliance on transferable private keys or mnemonic phrases and improving both security and usability. Together, these design choices demonstrate a practical approach to mitigating ticket scalping while preserving user privacy, and provide a foundation for future privacy-preserving ticketing platforms.

doi:10.6342/NTU202600625

Chapter 2 Related Works

2.1 Blockchain Ticketing Systems

2.1.1 Blockchain Technology

Blockchain was proposed by Satoshi Nakamoto in 2008 as the core data structure enabling Bitcoin. In essence, a blockchain can be viewed as a decentralized ledger replicated across a peer-to-peer network, where records are appended over time rather than modified retroactively.

As illustrated in Figure2.1, transactions are batched into blocks, and each block embeds a cryptographic digest of its predecessor, yielding a hash-linked sequence that makes undetected tampering computationally difficult.

To keep nodes consistent under asynchrony and adversarial interference, blockchain systems employ consensus protocols; early deployments commonly relied on Proof-of-Work, which supports eventual convergence to a single authoritative history under standard security assumptions.

Blockchain systems exhibit several fundamental properties:

- **Decentralization:** No single entity controls the ledger, reducing reliance on trusted

doi:10.6342/NTU202600625

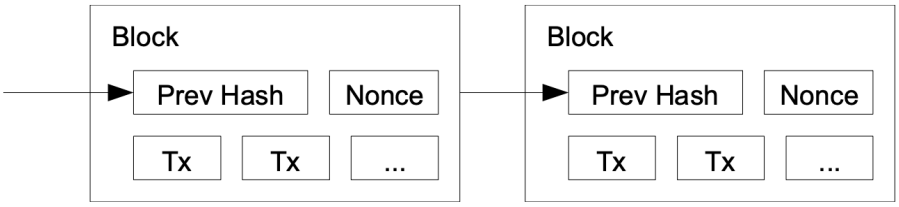


Figure 2.1: Basic structure of a blockchain (adapted from[27]).

intermediaries.

- **Immutability:** Once confirmed, historical records are computationally infeasible to alter without controlling a majority of the network's nodes.

- **Transparency:** Ledger data is publicly verifiable, enabling independent auditing.

Beyond cryptocurrencies, blockchain technology has been applied to wide range of domains, including digital marketplace, digital identity, decentralized finance (DeFi), voting systems, and ticketing platforms[7, 39].

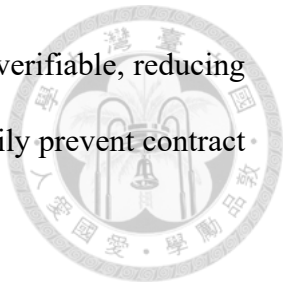
2.1.2 Ethereum and Smart Contracts

Ethereum was proposed by Buterin in 2013 and formally launched in 2015 as a general-purpose blockchain platform[5, 37]. Unlike Bitcoin, which primarily focuses on currency transfer, Ethereum introduces a Turing-complete execution environment that allows developers to deploy arbitrary application logic on-chain.

Decentralized applications (DApps) built on Ethereum differ from traditional centralized applications in several key aspects:

- **Fault Tolerance:** Decentralization allow the system to remain operational despite node failures or malicious participants.

doi:10.6342/NTU202600625



- **Trust Minimization:** Application logic and state are publicly verifiable, reducing reliance on platform operators. No single authority can arbitrarily prevent contract execution.

- **Composability:** Contracts enable interaction with each other and various clients, enabling modular and reusable financial and application primitives.

Compared to other blockchain platforms, Ethereum benefits from a large developer ecosystem, mature tooling, and extensive academic and industrial adoption. Its flexibility has led to widespread applications in decentralized finance, non-fungible tokens (NFTs), governance systems, and digital ticketing platforms[9, 36].

2.1.3 Blockchain Ticketing and Anti-Scalping Mechanisms

Blockchain-based ticketing systems have been widely studied as a promising solution to long-standing issues in conventional ticketing platforms, particularly ticket fraud, lack of transparency, and unfair ticket distribution. By leveraging the inherent properties of blockchain—immutability, transparency, and auditability—ticket issuance, ownership transfer, and redemption can be publicly verifiable and resistant to tampering. Early work by Tackmann demonstrated that blockchain can effectively prevent ticket duplication and counterfeit resale by enforcing unique ownership and verifiable transfer records on-chain, establishing a foundational security model for blockchain-based ticketing systems[30]. Beyond authenticity and traceability, recent research has increasingly focused on ticket scalping, where automated agents or intermediaries acquire tickets in bulk and resell them at inflated prices. Existing blockchain-based approaches address scalping through a variety of mechanisms, which can be broadly categorized as rule-based enforcement, resale

doi:10.6342/NTU202600625

regulation, identity binding, behavioral detection, and entrance control. Table2.1 summarizes several notable approaches.

A common approach to anti-scalping is to encode ticketing policies directly into smart contracts, ensuring that resale and transfer rules are enforced automatically and transparently. One widely adopted mechanism is **price-cap resale**, where the smart contract prohibits tickets from being resold above their original price. (e.g. Regner et al., Jiang et al. and DeTi[28, 29, 40]) Another related mechanism is **resale royalty or taxation**, where a portion of the resale value is automatically redistributed to the event organizer or platform. Li et al. introduce a taxation-based resale model that applies progressive fees to resale transactions[24]. Smart contracts are also used to impose **transfer limits** on NFT tickets. For example, Ma et al. explicitly limit the number of ticket transfers and require re-authentication upon each transfer, while Regner et al. restrict transferability through policy-based constraints embedded in NFT contracts[26, 29]. These mechanisms reduce repeated resales but do not prevent collusive off-chain transactions between specific buyers and sellers.

To address the limitation of seller-controlled resale, several systems introduce **controllable resale mechanisms**, where sellers are prevented from choosing specific buyers. This design aims to eliminate off-chain coordination and side payments. DeTi adopts a queue-based resale mechanism, where buyers and sellers are matched in a first-come-first-served manner, ensuring fairness and transparency in secondary marketscite[28]. Gysel et al. further propose a cryptographically protected resale market using shuffled and encrypted transactions, preventing sellers from identifying buyers during the resale process[20]. At the event entrance, many systems employ **QR code−based ticket verification** to prevent ticket sharing and duplication. Time-based or dynamically refreshed QR

doi:10.6342/NTU202600625

codes are commonly used to mitigate screenshot reuse and unauthorized transfers. For example, Tackmann introduces timestamped QR codes, while Ma et al. employ dynamic QR codes that are refreshed upon each scan and invalidated after use[26, 30]. Although effective against ticket sharing at entry time, QR-based mechanisms alone do not prevent unfair ticket acquisition or resale prior to the event.

Another line of work treats scalping as a behavioral anomaly detection problem. Systems such as PureNFT and the framework proposed by Ma et al. integrate **AI-based scalper detection models** that analyze user behavior—such as purchase frequency, IP patterns, or transaction timing—to identify and blacklist suspected scalpers[22, 26]. Once flagged, users may be prevented from purchasing or reselling tickets. While AI-based approaches can be effective in practice, they are inherently reactive and surveillance-driven, requiring extensive data collection and centralized decision-making.

To control bulk purchases and unauthorized transfers, many blockchain ticketing systems adopt **identity de-duplication**, similar in spirit to real-name ticketing systems. Tickets may be bound to real-world identities, platform accounts, or verified credentials. Liu proposes binding tickets to encrypted real identity numbers, while Ma et al. rely on government-issued real-name authentication integrated with consortium blockchains[25, 26]. DeTi similarly enforces identity verification through phone numbers or KYC services[28]. However, strong identity binding raises significant privacy concerns in blockchain-based ticketing systems. Most existing approaches protect sensitive identity data primarily through off-chain storage or encryption. While such techniques reduce direct exposure of personal information, ticket ownership and transaction records remain linkable across events and observable by platforms, consortium members, or regulators (e.g., Li et al.; Liu; Ma et al; PureNFT[22, 24–26]). While existing systems address scalping through

doi:10.6342/NTU202600625

rule enforcement, economic deterrence, identity binding, or behavioral surveillance, most approaches either sacrifice user privacy or rely on trusted intermediaries. This motivates the design of a privacy-preserving anti-scalping ticketing system that enforces fairness at the protocol level without real-name enforcement or behavioral monitoring.

Table 2.1: Blockchain Ticketing Mechanisms for Market Control

Mechanism	**Example**
Identity de-duplication	[11, 20, 24–26, 28, 40]
Price caps	[20, 28, 29, 40]
Resale royalty / taxation	[20, 24, 29, 40]
Transfer limits	[26, 29]
Authorized resellers / whitelist	[11] [40] [26]
Controlled resale	[20, 28]
AI-based scalper detection	[22, 26]
Dynamic entry credential	[26, 30]

2.2 Privacy-Preserving Identity Management

Identity management plays a central role in electronic ticketing systems, particularly when addressing both fairness and privacy requirements. a substantial body of research has explored privacy-preserving identity management mechanisms that aim to decouple ticket validity from direct identity disclosure. The techniques often employed include zero-knowledge proofs and decentralized identity frameworks.

2.2.1 Zero-Knowledge Proofs

Zero-knowledge proofs (ZKPs) enable proving a statement without disclosing other information[16]. This techniques are broadly used to preserve privacy in decentralized systems, by allowing parties to verify properties or correctness of computation without secret inputs[3].

doi:10.6342/NTU202600625

A particularly practical class of ZKPs is zkSNARKs (zero-knowledge Succinct Non-interactive ARguments of Knowledge). zkSNARKs provide succinct proofs and without interaction between the verifier and prover after the initial setup, making them well suited for on-chain verification in resource-constrained environments [18]. These properties are essential for scalable privacy-preserving applications on blockchain platforms such as Ethereum, where verification cost and transaction size are critical considerations [5].

2.2.2 Decentralized Identity and Self-Sovereign Identity

Decentralized Identity (DID) and Self-Sovereign Identity (SSI) prioritize user control, privacy, and interoperability in identity management. Without relying on centralized authorities, DIDs are typically associated with cryptographic keys and can be resolved to DID Documents containing public keys, service endpoints, and other metadata. SSI builds upon the DID framework by empowering individuals to manage their digital identities and credentials. In SSI systems, users can obtain verifiable credentials from trusted issuers and sent to verifiers without intermediaries. This model enhances privacy by allowing selective disclosure of attributes and reducing reliance on centralized identity providers.

2.2.3 Privacy-Preserving Ticketing Systems

This section reviews prior work on privacy-preserving identity management in ticketing systems, focusing on how different designs achieve anonymity, unlinkability, and accountability, and highlighting their limitations in addressing fairness-related threats such as ticket scalping.

doi:10.6342/NTU202600625

Verslype et al. present one of the earliest comprehensive studies on privacy-preserving electronic ticketing. Motivated by the privacy risks of electronic identity (eID)–based systems, they explicitly separate identity bootstrapping from ticket usage, using real-world identities only during an initial registration phase. Their design introduces two alternative constructions based on time-scoped pseudonyms and anonymous credentials. In the anonymous credential–based construction, users authenticate through zero-knowledge proofs and event-specific pseudonyms derived via one-way functions. This design ensures that ticket purchases and validations remain unlinkable across events, even when managed by the same organizer[33].

Building on early anonymous credential systems, Cha et al. propose a blockchain ticketing system that explicitly targets unlinkability through non-interactive zero-knowledge proofs (NIZK). In their design, users generate fresh cryptographic randomness and one-time key material for each ticket purchase and validation, ensuring that different tickets owned by the same user cannot be correlated on-chain. Ticket servers submit transactions on behalf of users, further reducing metadata leakage[8].

Similarly, Han et al. introduce a privacy-preserving ticketing system based on attribute-based credentials (ABC). Users obtain anonymous credentials containing certified attributes and later prove ticket ownership and eligibility through zero-knowledge proofs with selective disclosure. By avoiding persistent identifiers and allowing fresh proofs for each interaction, the system guarantees unlinkability across multiple events and sessions, even against colluding issuers and verifiers. These works demonstrate that strong cryptographic privacy can be achieved without relying on platform accounts or real-name registration[21].

More recent studies explore self-sovereign identity (SSI) as a privacy-oriented alter-

doi:10.6342/NTU202600625

native to centralized identity management. Feulner et al. investigate the usage of SSI in event ticketing system, where tickets are issued as verifiable credentials and stored in user-controlled identity wallets. Through verifiable presentations, users can selectively disclose required attributes while keeping personal data off-chain and under their control. From a privacy perspective, SSI-based ticketing improves data minimization and user autonomy compared to real-name systems[14].

Zhan et al. propose PriTKT, a blockchain-based privacy-preserving ticketing system designed for IoT environments. PriTKT integrates attribute-based anonymous credentials, unlinkable signatures, and zero-knowledge proofs to provide unlinkability and conditional traceability. Blockchain is used as an immutable ledger to support double-spending detection and auditability, while user identities remain hidden unless malicious behavior is detected[41].

Similarly, Lafourcade et al. present a transferable and auditable ticketing protocol that achieves anonymity and unlinkability with strong formal proofs. Their system enables ticket purchase, validation, transfer, and refund without revealing persistent identifiers, while allowing conditional deanonymization through a trusted judicial authority. These works demonstrate that anonymity, unlinkability, and accountability can coexist within rigorously defined cryptographic frameworks[23].

Across existing privacy-preserving ticketing systems, many works successfully achieve anonymity, selective disclosure, and unlinkability across events through anonymous credentials or zero-knowledge proofs. However, these designs largely focus on ticket authenticity and access control, and rarely integrate mechanisms to enforce fairness in adversarial market conditions. Conversely, systems that actively address scalping and resale

doi:10.6342/NTU202600625

manipulation often rely on strong identity binding, behavioral surveillance, or regulatory oversight, sacrificing unlinkability in the process.

2.2.4 Semaphore Protocol

Semaphore is a zero-knowledge protocol designed for Ethereum that enables users to anonymously prove membership in a group and broadcast signals without revealing their identities.

2.2.4.1 Identity

In Semaphore, each user generates an identity locally to used as unique identifier. An identity consists of a pair of EdDSA keys and a commitment. The commitment acts as the public representation of an identity in groups, instead of exposing the public key directly. An identity can be restored by importing the secret key.

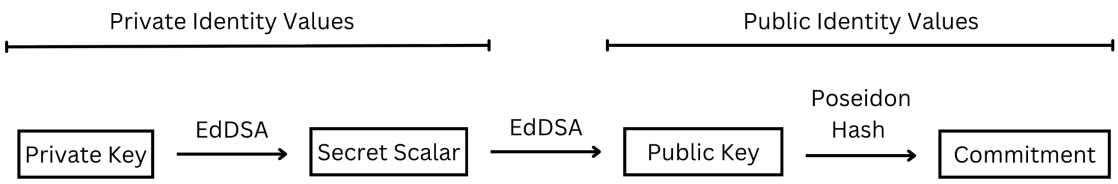


Figure 2.2: Semaphore V4 identities, adapted from[34]

2.2.4.2 Group

A Semaphore group is a Merkle Tree in which each leaf is an identity commitment for a user. Given two sibling nodes L_i and L_{i+1}, their parent node is computed as:

$$P = \text{Hash}(L_i \parallel L_{i+1})$$

doi:10.6342/NTU202600625

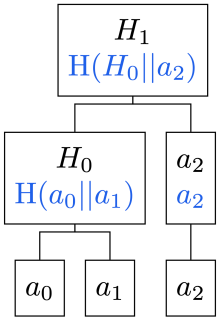


Figure 2.3: Merkle tree structure, adapted from[34]

Whenever a member is added or removed, the corresponding leaf is updated and all affected hashes along the path to the root are recomputed, resulting in a new Merkle root.[19].

The group is based on a leanIMT structure, which supports dynamic tree depth and avoids storing zero-padding leaves as well as unnecessary internal nodes, enabling efficient on-chain updates and verification. The complexity of insertion and removal operations is $O(\log N)$, where N is the number of leaves[43].

2.2.4.3 Proof

To anonymously broadcast a signal, a user generates a zkSNARK proof containing the following components:

Nullifier The nullifier is used to prevent double signaling within the same context. It is computed as:

$$\text{nullifierHash} = \text{Hash}(\text{identityNullifier}, \text{externalNullifier})$$

The nullifier hash is published on-chain and checked for uniqueness. Because it depends on the external nullifier, the same identity can be reused across different contexts without linkability[13].

doi:10.6342/NTU202600625

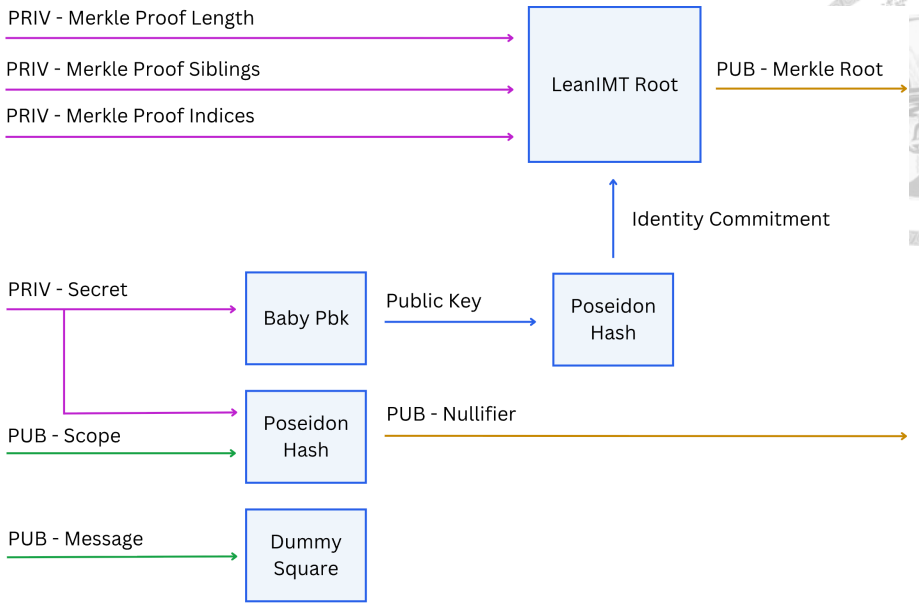


Figure 2.4: Semaphore V4 proof verification circuit, adapted from[34]

Signal The signal represents the message the user wants to broadcast (e.g., a vote or confirmation). The signal is hashed and included as a public input to the zkSNARK circuit, binding the proof to the message without revealing the sender' s identity[13].

Scope (External Nullifier) The scope, also referred to as the external nullifier, defines the context in which double-signaling prevention applies. By incorporating the external nullifier into the nullifier hash, Semaphore ensures that nullifiers are unique per context, such as a specific event, poll, or ticketing instance[13].

Merkle Proof To prove group membership, the user includes a Merkle proof consisting of sibling hashes along the path from their identity commitment to the Merkle root. Within the zkSNARK circuit, these hashes are recomputed step by step and compared against the public Merkle root stored on-chain. If the computed root matches, the proof confirms valid membership without revealing which leaf was used

doi:10.6342/NTU202600625

2.2.5 Proof of Personhood

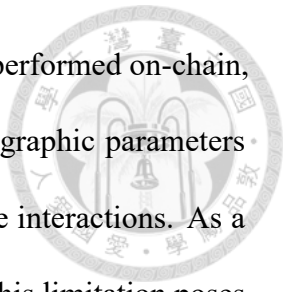
The concept of Proof of Personhood (PoP) was introduced to address fundamental challenges in decentralized systems where participation is permissionless and identities are cheap to create. In such environments, adversaries can easily launch Sybil attacks by generating multiple pseudonymous identities, undermining fairness mechanisms such as voting, airdrops, or resource allocation [6]. PoP aims to enforce a one-person-one-identity property while avoiding reliance on real-world identity disclosure, thereby balancing fairness and privacy.

World ID is a practical realization of the PoP concept that leverages zero-knowledge proofs to enable privacy-preserving uniqueness verification. Users enroll by visiting a dedicated hardware device called an Orb, which performs biometric verification (primarily iris scanning); the biometric data is processed locally and transformed into an irreversible representation used only to check uniqueness, while raw biometric images are not stored.

After enrollment, users manage their credential through the World App, which allows them to generate proofs when needed. During verification, World ID relies on Semaphore to produce a zero-knowledge proof that confirms the user is a valid, unique human and has not previously used the same credential, while emitting a non-linkable nullifier to prevent double use. Context-specific nullifiers ensure that a user can only successfully prove personhood once per application or action.

One of the key advantages of World ID is its compatibility with blockchain environments such as Ethereum. Proofs generated by World ID can be verified on-chain using smart contracts [38], this makes World ID particularly suitable for on-chain access control and anti-Sybil mechanisms.

doi:10.6342/NTU202600625



However, while verification of Semaphore-based proofs can be performed on-chain, the generation of proofs requires the prover to have access to cryptographic parameters and the latest group state, which are typically obtained through online interactions. As a result, World ID does not natively support offline proof generation. This limitation poses challenges for use cases such as event entrance verification, where network connectivity may be unavailable or unreliable, motivating the need for complementary offline-capable authentication mechanisms.

2.3 Passkey-Based Authentication

Passkeys are a modern, passwordless authentication mechanism standardized by the World Wide Web Consortium (W3C) and the FIDO Alliance through the WebAuthn and FIDO2 specifications [15, 35]. They were proposed to address long-standing security and usability issues of password-based authentication, such as phishing, credential reuse, and database breaches. Instead of shared secrets, passkeys rely on public-key cryptography and device-assisted user verification.

At a high level, a passkey consists of a cryptographic key pair generated locally on the user's device. During registration, only the public key is exposed to the relying party. The private key are secretly stored on the client device. Authentication is performed through a challenge–response protocol: the server issues a random challenge, and the client produces a signed assertion using the private key[35]. This process is non-phishable, as the private key never leaves the device and signatures are bound to the requesting origin.

Passkeys are typically stored in secure, hardware-backed environments provided by the operating system, such as the Secure Enclave on Apple devices or Trusted Platform

doi:10.6342/NTU202600625

Modules (TPMs) on other platforms. Access to the private key is gated by local user verification mechanisms, including biometrics (e.g., fingerprint or face recognition) or device unlock credentials. Applications interact with passkeys through platform APIs, and private keys cannot be directly accessed or exported by applications.

Major platform providers, including Apple, Google, and Microsoft, support passkeys natively across their ecosystems [1, 17]. Passkeys can be managed through platform-specific credential managers and, in some cases, synchronized across a user's devices via encrypted cloud services. This synchronization improves usability by allowing users to authenticate on new devices without re-registration. For user concerns regarding centralized storage, passkeys can also be stored and managed locally with password managers or hardware security keys.

Passkeys are increasingly adopted in various applications, including web authentication, mobile applications, and device login. They are particularly suitable for scenarios that require strong user authentication with minimal friction, such as account login, transaction confirmation, and device-bound authorization flows.

Despite their advantages, passkeys have several limitations. Support depends on platform and browser compatibility, and older devices may not fully support WebAuthn-based authentication. Besides, passkeys do not inherently guarantee uniqueness at the human level, as a single user may create multiple passkeys across different devices or accounts. Passkey synchronization mechanisms allow credentials to be transferred between devices within the same ecosystem, which may enable ticket transfer under certain conditions. Therefore, passkeys are used in our system as a device-bound authentication factor, along with verification of proof of personhood.

doi:10.6342/NTU202600625

Chapter 3 System Design

This section presents the overall design of the proposed ticketing system BlozkTix, focusing on how privacy preservation and fairness enforcement are achieved throughout the ticket lifecycle. The system is designed to address common challenges in online ticketing, including ticket scalping and privacy risk.

The following subsections describe the design of identity management, ticket purchase, seat selection, refund, and entry mechanisms in the system.

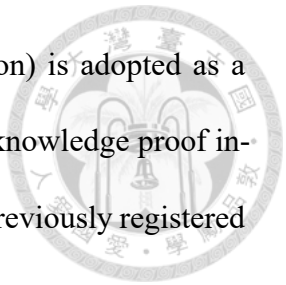
3.1 Lifecycle-scoped Identity Structures

The system employs multiple identities as shown in Figure 3.1, each serving a distinct purpose within the ticket lifecycle.

Figure 3.1: Lifecycle-scoped identity structures

- **Passkeys** as the root of trust in the system, they are used to derive seeds for creating and loading user identities.

doi:10.6342/NTU202600625



- **Decentralized identity** (e.g. World-ID used in implementation) is adopted as a proof-of-personhood mechanism. It is used to generate a zero-knowledge proof indicating that a user represents a unique individual who has not previously registered on the platform.

- The **platform identity** represents a persistent user identity shared across multiple events on the platform. This identity is used to generate event-scoped proof, which demonstrates that the user has not previously purchased a ticket for a specific event.

- A distinct **event identity** is generated for each event. This design prevents direct linkage between real-world behaviors (such as seat selection and venue entry) and the transaction record. This event identity is then used to generate proof for anonymously seat ownership and entrance. It is generate from same passkey with platform identity to prevent transfer in isolation.

By separating identities from context-specific proofs, the system enforces usage constraints while preventing unnecessary linkage between user actions.

3.2 Presale Configuration

Before ticket sales begin, the administrator configures the event parameters through the smart contract. The configuration includes the ticket sale start and end time, total ticket supply, resale queue capacity, and ticket price. All parameters are stored on-chain to ensure transparency and immutability.

doi:10.6342/NTU202600625

3.3 Platform Signup

Before participating in any event, users are required to complete a signup process to establish a platform level identity that can be reused across multiple events. The process of registration is shown in Figure 3.2.

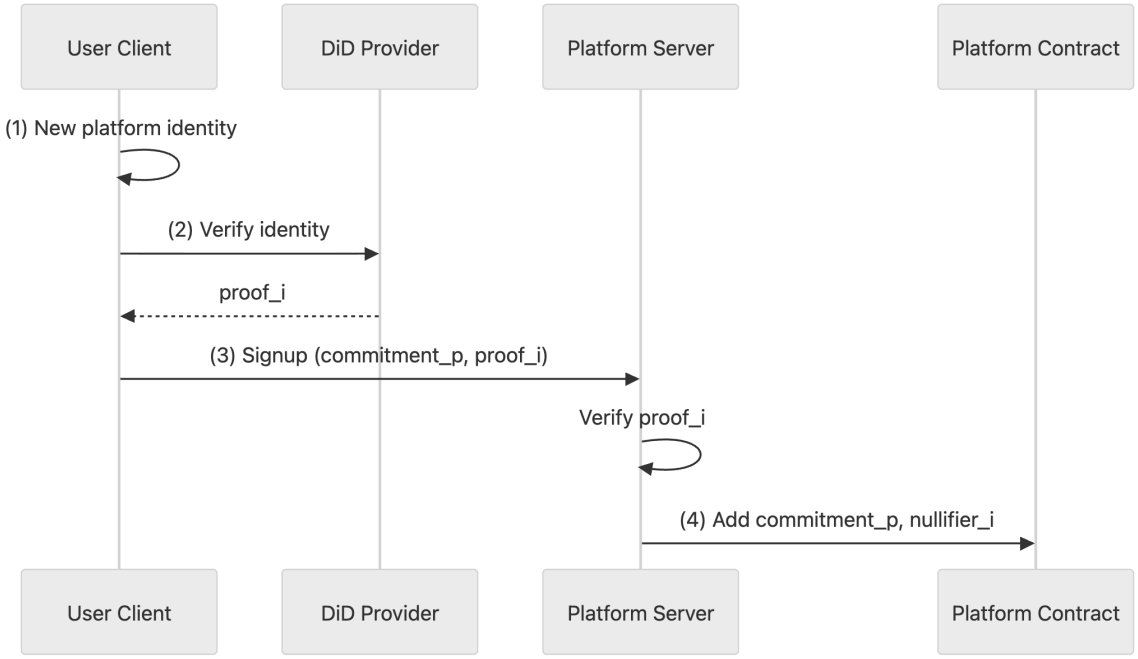


Figure 3.2: Platform signup flow

1. During signup, a user first creates a local identity using a passkey-based authenticator.

2. The user then generates a decentralized identity (DiD) proof to demonstrate uniqueness.

3. The proof is verified by the server to ensure that each real person can register only once.

4. Upon successful verification, the corresponding DiD provider nullifier is bound to the platform identity within the platform contract, preventing duplicate registrations

doi:10.6342/NTU202600625

while preserving user anonymity. At the same time, the commitment is added as a platform member.

This identity is used during ticket purchase and event entrance to prevent repeated purchases and to ensure that the same ticket holder completes the entry process. Subsequent interactions with event contracts rely on zero-knowledge membership proofs derived from this platform identity, eliminating the need to repeatedly verify DiD and enabling privacy-preserving and offline-capable ticket usage in later stages.

The verification of DiD proof during signup is performed off-chain to reduce gas cost and to allow flexibility in integrating different DiD providers. To prevent manipulation, the platform contract records all registered DiD nullifiers on-chain, enabling public auditability of the registration process.

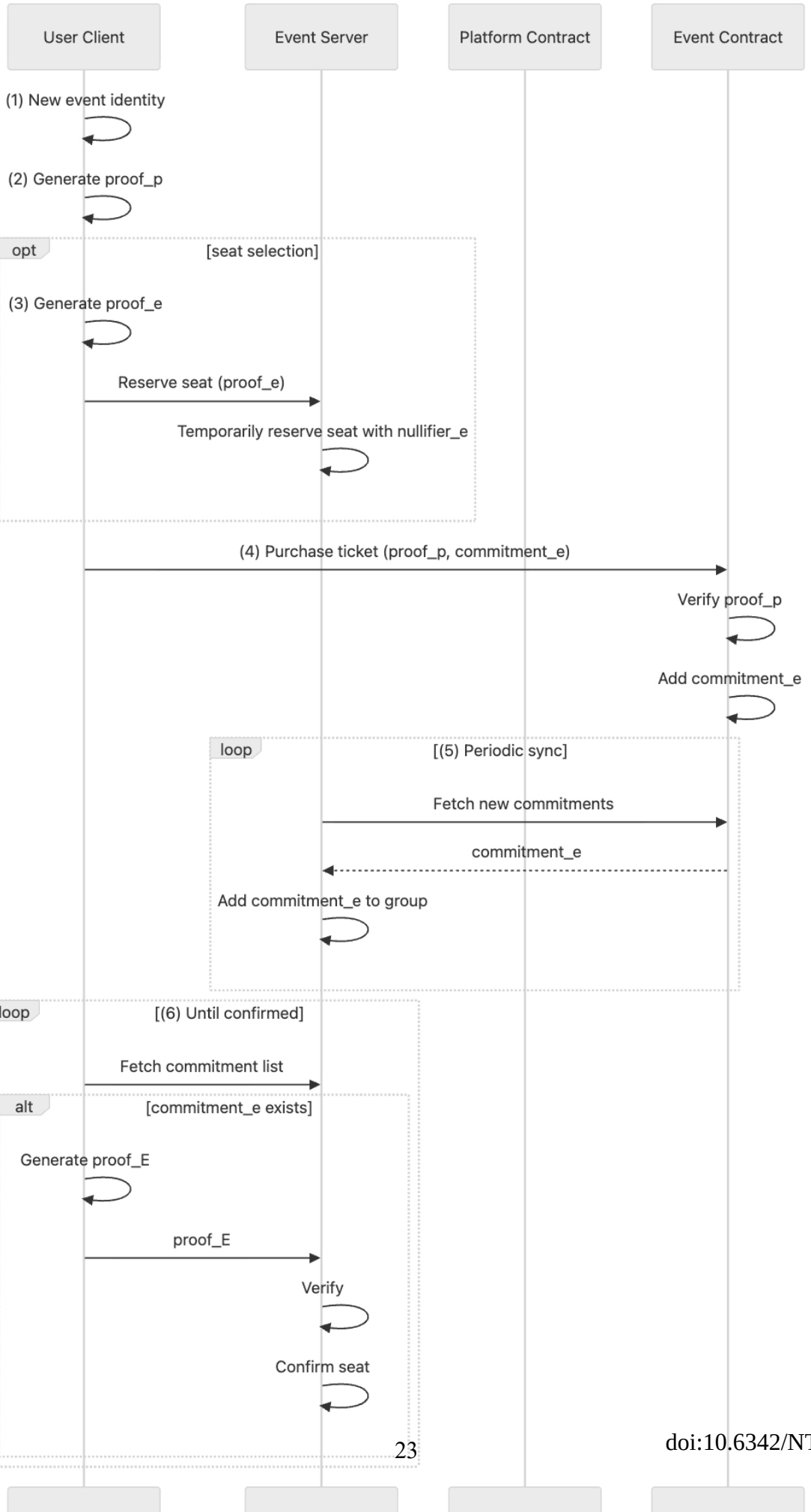
3.4 Ticket Purchase

The ticket purchase process integrates decentralized identity verification, optional anonymous seat selection, and on-chain payment. Figure 3.3 illustrates the overall flow.

1. The user generates a event Semaphore identity locally on the client device. Only the identity commitment is published on-chain, representing ticket ownership.

2. To enforce a one-ticket-per-member policy, a platform level proof is required during ticket purchase. The scope of this proof is unique to each event. The purchase proof includes the user’ s identity commitment, ensuring that the proof is bound to the ticket identity. Verification is performed on-chain to guarantee transparency and prevent manipulation.

doi:10.6342/NTU202600625



doi:10.6342/NTU202600625

3. If seat selection were performed directly on-chain using a wallet transaction, seat choices could be linked to the user's identity. To avoid this privacy risk, all seat-related operations are processed by the server. Before payment, a user may optionally reserve a preferred seat with event proof nullifier. Users may also skip seat selection entirely and proceed directly to payment for faster checkout. After purchase, seats can be changed freely among available options until finalization.

4. Payment is executed through an on-chain transaction to ensure immutability and public verifiability. The user submits the purchase proof along with their event commitment to join the ticket-holder group.

5. Once the purchase transaction is confirmed, the server synchronizes the new event commitment into a event Semaphore group. This enables users to later prove ticket ownership anonymously without revealing their wallet address or platform identity.

6. User confirm the seat ID registration with a valid proof. The seat reservation expires automatically if not confirmed within time window.

3.5 Refund Ticket

Since tickets are non-transferable, ticket holders who no longer wish to attend the event may request a refund through the system. The refund process is shown in Figure 3.4.

1. To be eligible for ticket cancellation and refund, the user must first clear any associated seat ownership and mark the ticket as spent. Because seat selection is performed anonymously, the user is required to generate event proof again to demon-

doi:10.6342/NTU202600625

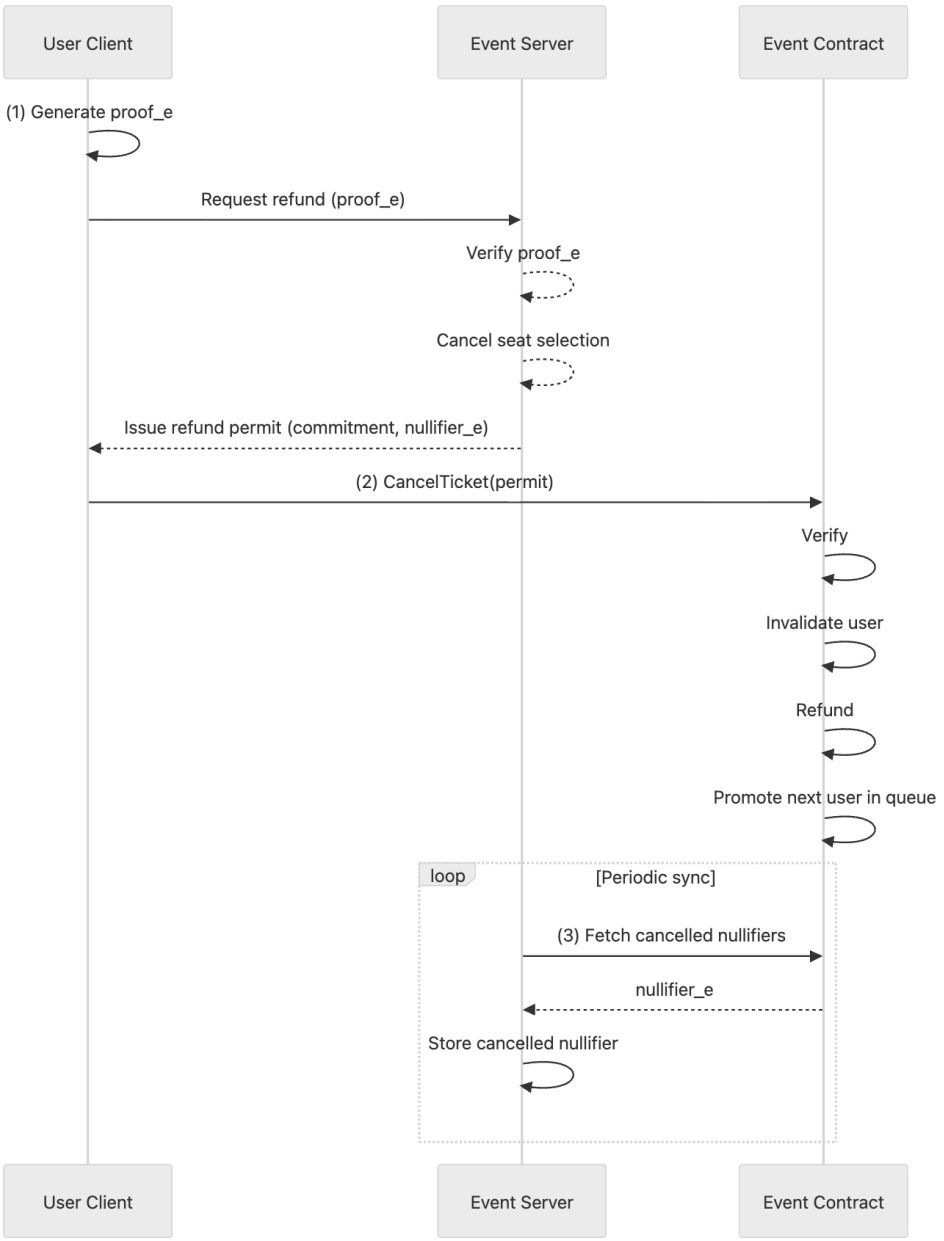


Figure 3.4: Ticket refund process

doi:10.6342/NTU202600625

strate valid seat ownership and to indicate their intention to relinquish the ticket.

After verifying the proof, the server issue a refund permit to the user. This permit serves as cryptographic authorization that the ticket is eligible for cancellation and refund.

2. The user initiates ticket cancellation by submitting the refund permit to the smart contract. The refund permit consists of the ticket commitment and the corresponding event nullifier, and is signed by the administrator's wallet. Upon successful signature verification, the contract invalidates the user's wallet address and purchase nullifier, thereby allowing repurchase using the same wallet and platform identity in the future. The ticket price is then refunded to the user's wallet. After the refund is completed, the contract triggers the ticket resale mechanism.

3. After observed ticket cancellation events from the contract, server add the event nullifier, to prevent a ticket spent at entrance after cancellation.

3.6 Queue

Ticket resale is implemented by promoting users from a queue to replace former ticket holders when cancellations occur. The queue follows a first-come, first-served policy and preserves chronological order.

User can join the queue when ticket sold out. The process is similar to purchase except of seat selection are omitted. In current design, user need to pay the full amount of the ticket price, so that the resale can perform smoothly.

The process to join a queue is illustrated in Figure 3.5

doi:10.6342/NTU202600625

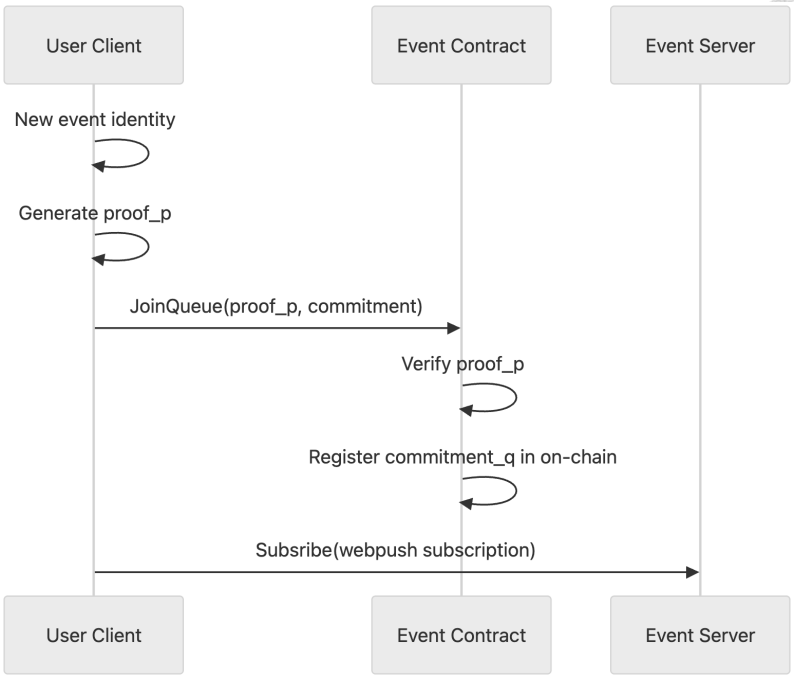


Figure 3.5: Queue process

When a user is successfully promoted from the queue to become a ticket holder, they are notified via web push. The new ticket holder may freely select any available seat that has not yet been registered, and is not restricted to the seat previously held by the refunded ticket holder. This typically occurs when the former holder had not finalized seat selection. During the remaining sale period, the new ticket holder may also change their seat selection. Figure 3.6 illustrates the promotion process.

Users may cancel their queue participation at any time during the sales period, provided they have not yet been promoted to ticket holder status. Upon removal from the queue, the prepaid ticket price is refunded to the user's wallet. After the sale period ends, any remaining users in the queue are batch-cancelled and refunded. The queue cancellation process is demonstrated in Figure 3.7.

doi:10.6342/NTU202600625

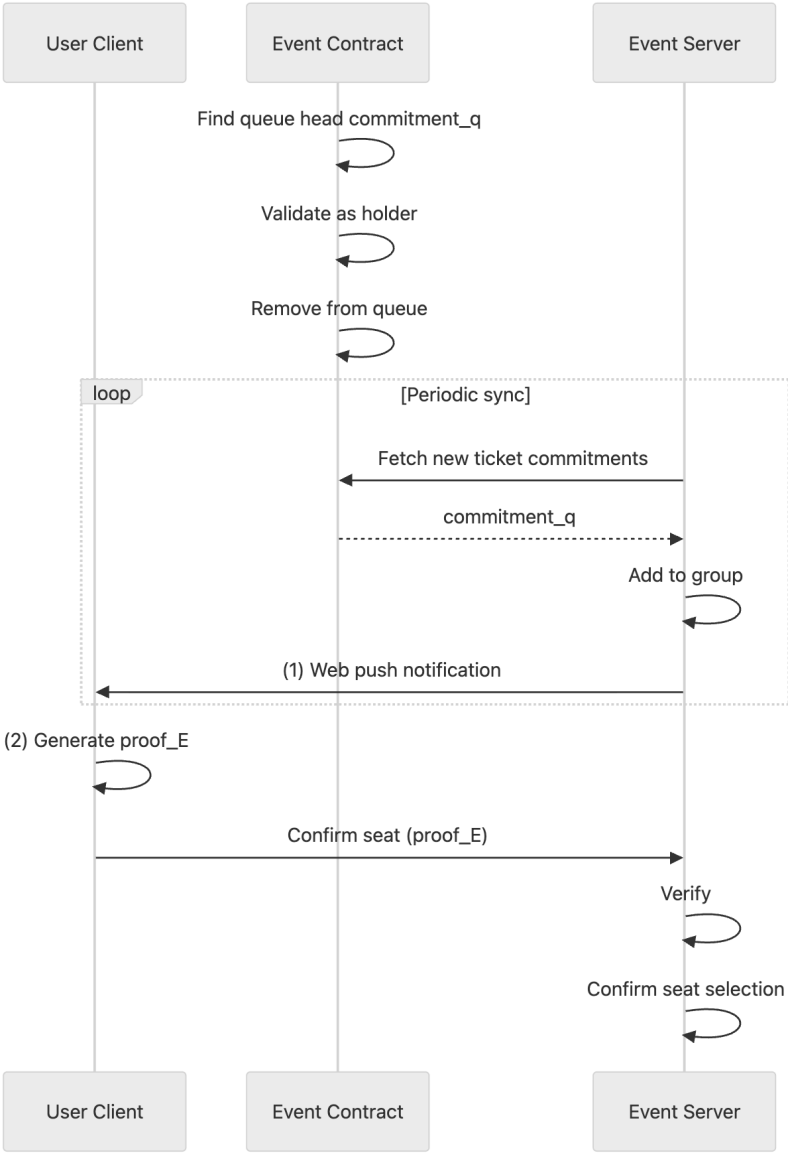


Figure 3.6: Queue promotion process

doi:10.6342/NTU202600625

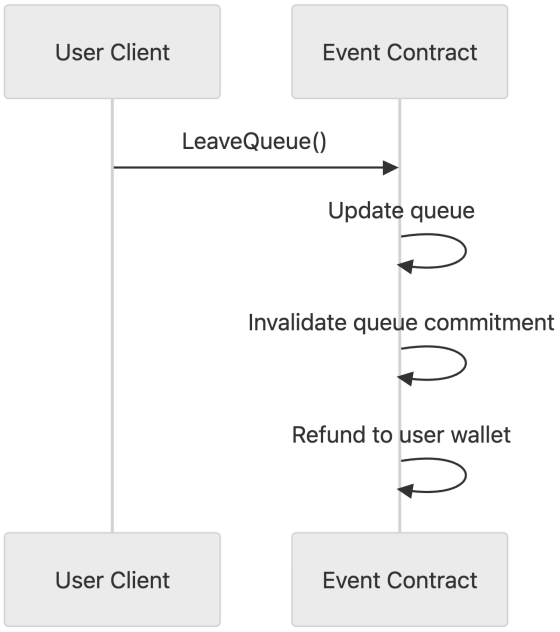


Figure 3.7: Queue cancellation process

doi:10.6342/NTU202600625

3.7 Event Entry

Event entry is verified through zero-knowledge proofs to ensure that only valid ticket holders can gain access, while preserving their anonymity. The entry verification process is illustrated in Figure 3.8.

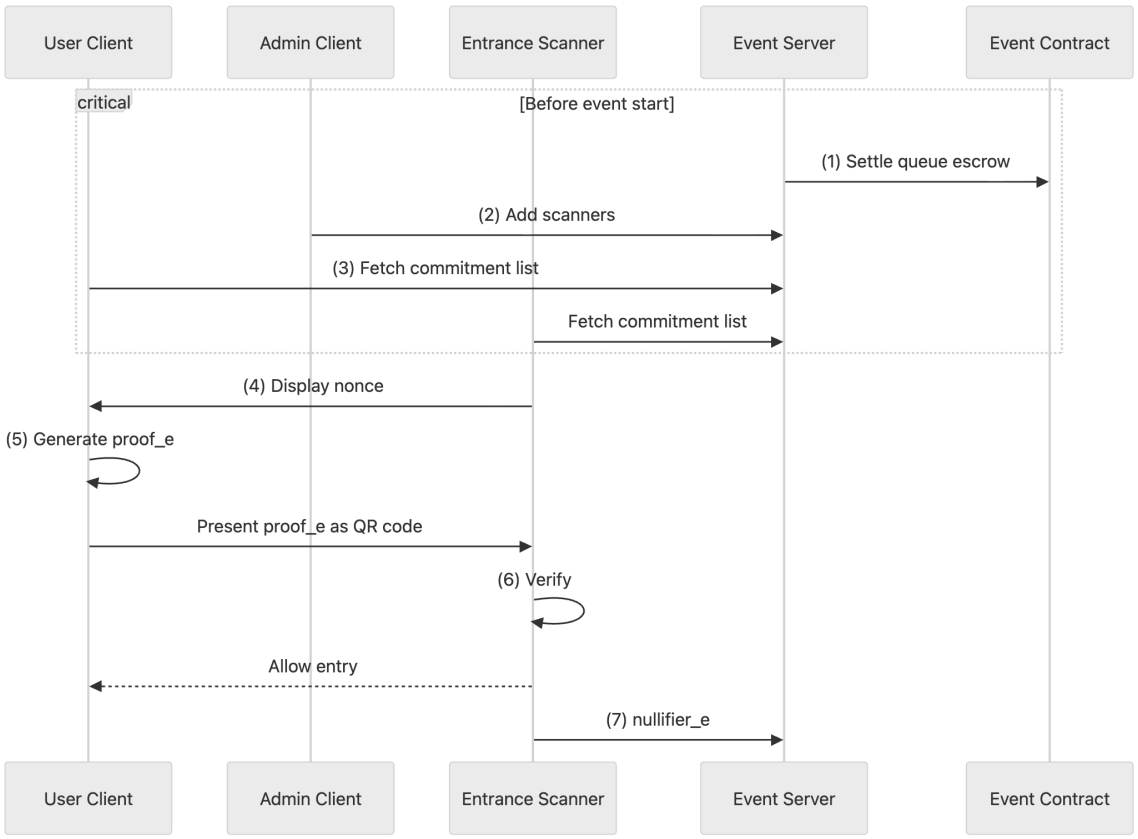


Figure 3.8: Event entry and seat verification flow

1. After ticket sales conclude, the administrator returns the funds to remaining users in queue.

2. Wallet addresses used by scanners are also registered with the server.

3. To enable offline proof generation and verification, users and scanner must download the finalized list of group commitments from the server.

doi:10.6342/NTU202600625

4. At the event entrance, scanners are responsible for verifying tickets at the venue entrance. Each scanner generate a short-lived nonce and present it to the user.

5. The user generates an entry proof bound to this nonce using the downloaded commitment list. The proof is presented as a QR code.

6. Proof verification is performed locally by the scanner.

7. Upon successful verification, the scanner submits the nullifier to the server to mark the ticket as spent.

Throughout this process, the user’ s identity remains undisclosed.

3.8 Finalization and Settlement

After the event concludes, the system enters the finalization and settlement phase to securely complete ticket usage and fund distribution. During this phase, entry nullifiers collected during on-site verification are uploaded to IPFS and hash are recorded on the smart contract to ensure immutability and public auditability.

After settlement is completed, the event organizer is allowed to withdraw the finalized funds from the escrow. This separation between entry finalization and fund withdrawal ensures that payments are only released after ticket usage has been cryptographically confirmed, providing fairness and auditability without requiring direct linkage between users and their real-world identities.

doi:10.6342/NTU202600625

3.9 Execution Layers and Design Rationale

The system adopts a hybrid execution model that carefully separates on-chain and off-chain responsibilities to balance transparency, fairness, and usability. The separation is summarized in Table 3.1.

Operation	Execution Layer	Rationale
Ticket Purchase	On-chain	Transparency, fairness enforcement
Queue Management	On-chain	Fairness, public verifiability
Refund and Resale	On-chain	Fairness, auditability
Seat Selection	Off-chain	Privacy preservation, usability
Entrance Verification	Off-chain	Privacy preservation, offline capability

Table 3.1: Separation of execution layers

Core operations related to ticket purchase, queue management, refund, and resale are handled on-chain to ensure transparency and enforce fair ticket distribution according to predefined policies. By executing these mechanisms on-chain, the system enables public verifiability and prevents manipulation of ticket allocation results.

In contrast, seat assignment and entrance verification are handled off-chain for unlinkability between real-world behaviors and user wallets identities. Off-chain execution also enables a smoother user experience at the venue, supporting offline entrance verification and eliminating waiting time for on-chain transactions. Additionally, handling minor interactions off-chain allows users to complete common actions without paying gas fees. This design reduces user friction and improves accessibility, while maintaining strong security guarantees through cryptographic proofs and controlled verification.

doi:10.6342/NTU202600625

Chapter 4 System Implementation

Figure 4.1: System architecture overview

While BlozkTix is designed as a ticketing platform that can support multiple events, the implementation presented in this thesis focuses on a single-event prototype to reduce complexity and to allow clearer evaluation of the system's primary privacy and fairness

doi:10.6342/NTU202600625

properties. Hence, we merge the process of platform registration and purchase ticket are used directly as unique user identifiers.

As shown in Figure 4.1, the system is composed of three major components: a smart contract deployed on Ethereum-compatible blockchains, a backend server responsible for synchronization and off-chain logic, and frontend clients for users, scanners, and administrators. The implementation emphasizes robustness, gas efficiency, and compatibility with zero-knowledge proof workflows.

4.1 Event Contract

The event contract implements the core ticket lifecycle, including purchase, queuing, refund, and final audit. The contract is developed using Solidity to be deployed on Ethereum-compatible blockchains.

4.1.1 Core Data Structures

The contract maintains several critical data structures:

As shown in Figure 4.2, the main data structures include:

- Purchase nullifier, preventing multiple purchases using the same platform identity.
- Queue linked list for FIFO promotion.
- Event configurations, holder count and queue size to enforce sale limits.
- Audit CID, an IPFS hash of json data including finalized seat assignments and entry-related nullifiers.

doi:10.6342/NTU202600625

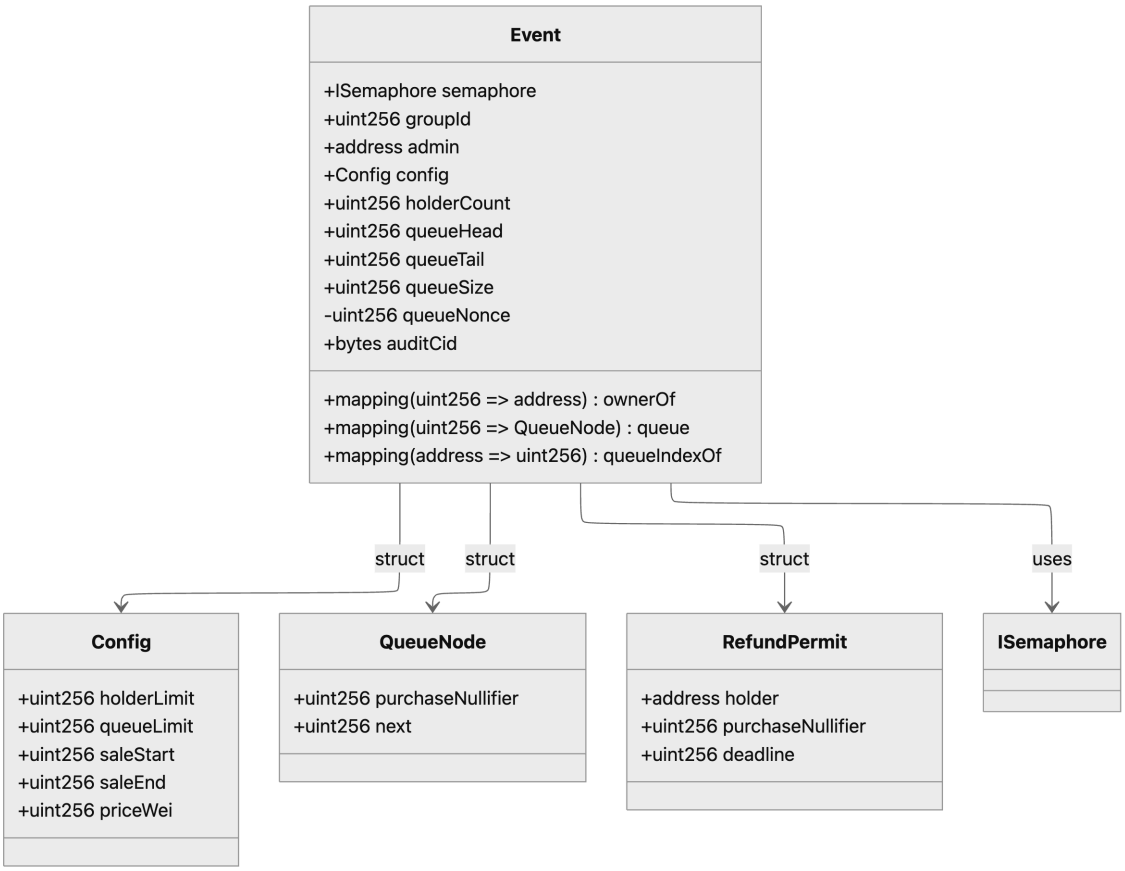


Figure 4.2: Smart contract data structures

All core data structures are designed to minimize gas consumption. Only essential state is stored in the contract, while auxiliary data are managed by off-chain server or IPFS.

4.1.2 Contract Functions

The contract exposes several functions to manage the ticket lifecycle. The key functions are summarized below in Table 4.1.

doi:10.6342/NTU202600625

Table 4.1: Smart Contract Functions

Function	Description
purchaseTicket	The contract allows a user to purchase a ticket by submitting a purchase proof, payment and event identity commitment to used in zero-knowledge group. After successfully verified the purchase proof, the payment is accepted and the commitment is added as a holder. If tickets are sold out, the user is added to the queue. The purchase nullifier is recorded to prevent multiple purchases.
cancelTicket	Enables a ticket holder to request a refund by submitting an admin-signed refund permit and a spent nullifier. The permit certifies that the user has relinquished any seat ownership and has generated a valid spent nullifier. The user's wallet is invalidated and purchase nullifier is cleared to allow repurchase. Then, the queue head is promoted as a ticket holder if applicable.
cancelQueue	Allows a queued user to cancel their position in the queue and receive a refund. The corresponding user is removed from the queue linked list upon cancellation. This logic is intentionally isolated from ticket cancellation logic.
finalizeSeat	Enables the administrator to batch finalize seat assignments by mapping seats to nullifiers for on-chain auditability.

Continued on next page

doi:10.6342/NTU202600625

Function	Description
finalizeEntry	Allows the administrator to batch record spent nullifiers for event entry, preventing double entry for on-chain auditability.
settleEscrow	Allows the administrator to process any remaining queued users by refunding them first.
withdrawFunds	Enables the administrator to withdraw accumulated funds after the event concludes.

Several design optimizations are applied to improve gas efficiency and user experience:

- The purchase function combines ticket payment and queue enrollment into a single operation. This design addresses a race condition where tickets may sell out between the moment a transaction is broadcast and when it is confirmed on-chain.This approach avoids transaction reverts due to temporary oversubscription and improves user experience while preserving fairness.
- Queue cancellation and ticket cancellation are implemented as separate functions. Ticket cancellation, in contrast, requires verification of an admin signature and spent nullifier checks and is therefore more expensive. By splitting these paths, the system avoids imposing unnecessary costs on queued users and improves scalability.
- Batch processing significantly reduces transaction overhead and allows administrators to finalize registrations and upload entrance data efficiently, which is particularly important for events with many participants.

doi:10.6342/NTU202600625

- The contract holds ticket payments in escrow until refunds are processed. The contract processes any remaining queued users by refunding them before releasing funds, ensuring that all participants are treated fairly. The event organizer only allowed to withdraw accumulated funds after the event concludes.

4.1.3 Ticket Identity Lifecycle

Figure 4.3 summarized the identity lifecycle in the contract.

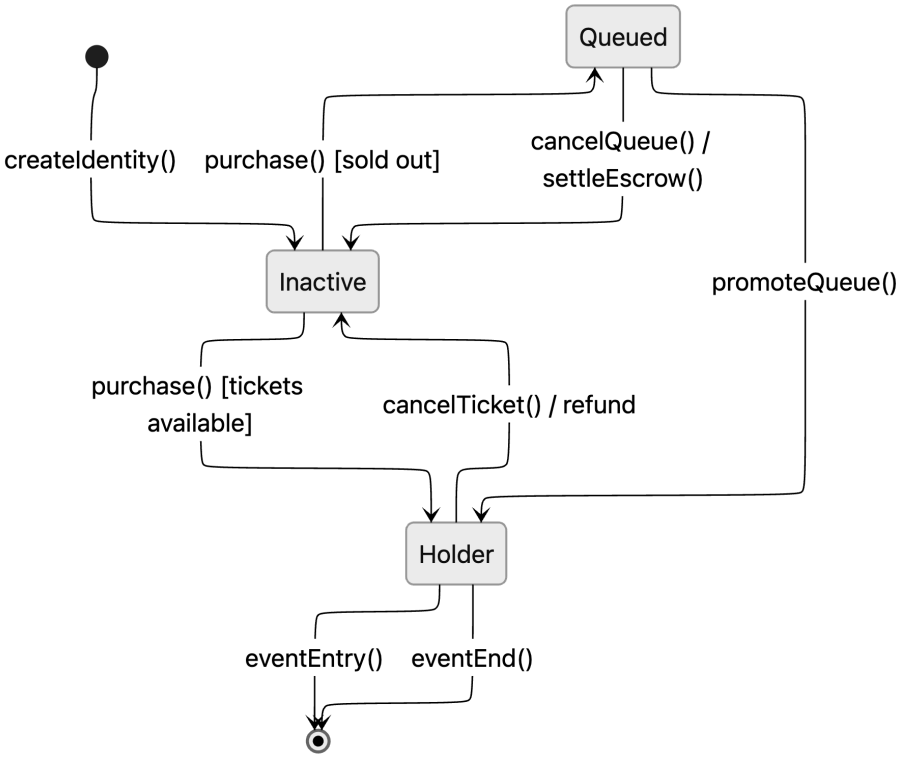


Figure 4.3: Identity lifecycle states

- **Inactive:** The identity has no active ticket and its purchase nullifier and the address used are cleared making them available for next purchase.

- **Holder:** The identity represents an active ticket owner and is eligible for seat selection and event entry. The nullifier is recorded to prevent repurchase until cancellation and refund.

doi:10.6342/NTU202600625

- **Queued:** The identity has paid for a ticket and is waiting in a deterministic FIFO queue for promotion. The address is linked to the identity until cancellation or promotion.

4.2 Backend Server

The backend server coordinates off-chain logic, state synchronization, and proof verification. It is implemented using Node.js and Next.js. As shown in Figure 4.1, the architecture consists of four main layers: a blockchain listener, an API layer, a service layer, and a storage layer.

4.2.1 Blockchain Listener

The listener component monitors on-chain state changes using viem. Instead of relying on event subscriptions, the system uses polling-based block synchronization. Polling is more resilient to network interruptions and provider instability, which are common in long-running event systems. The listener periodically:

1. Fetches new blocks,
2. Processes relevant contract events,
3. Updates backend services with newly confirmed commitments and spent nullifiers.

To ensure correctness, the listener maintains and verifies the latest synchronized block hash, guaranteeing that holder and refund states remain consistent with on-chain data.

doi:10.6342/NTU202600625

4.2.2 Services and API Layer

As shown in Figure 4.1, the server interacts with clients and contract-listener through an API layer, which is handled by multiple services, each responsible for a specific domain of functionality. Table 4.2 summarizes the backend API endpoints grouped by their respective services.

Table 4.2: Backend Services and API endpoints

Services	**Handling APIs**

SeatService
Manages seat reservations and confirmations.
	`POST /ticket/reserveSeat`
	`GET /ticket/getUnavailableSeats`
	`POST /ticket/getSeatsByNullifier`
	`POST /ticket/confirmSeat`
	`POST /ticket/changeSeat`

TicketService
Handles ticket-related operations for users.
	`GET /ticket/getCommitments`
	`GET /ticket/getSpentByNullifiers`
	`POST /ticket/addSubscription`
	`POST /ticket/requestRefund`
	`POST /listener/handleSync`

EntranceService
Handles scanner-side auth and entry tracking
	`GET /scanner/getNonce`
	`POST /scanner/login`
	`GET /scanner/refresh`
	`POST /scanner/addEntry`
	`POST /scanner/checkEntry`

AdminService
Manages admin-only operations.
	`POST /admin/sendAnnouncement`
	`POST /admin/getRegistrationData`
	`POST /admin/addScanners`

The api path is grouped by consumed client, which is either user(ticket), scanner, or admin. This separation improves modularity and limits the impact of changes in individual

doi:10.6342/NTU202600625

components.

4.2.3 Backend Storage Management

The server use a semaphore group and an off-chain database to complement on-chain state while avoiding unnecessary exposure of user identities. At the same time, these off-chain storage solutions also support efficient operations and reduce gas fees and time costs associated with on-chain transactions.

Semaphore Group Management The backend maintains the Semaphore group (which explained in Section 2.2.4.3) by aggregating commitments derived from on-chain events polled and forwarded by the listener (Section 4.2.1) with API to TicketService (Section 4.2.2). The group can directly used by backend for verification, while clients need to fetch the latest full commitment list to construct a local Merkle tree for proof generation. This design avoids revealing individual membership changes and ensures that proof verification remains consistent across clients.

Off-chain Database In addition to on-chain state, the system relies on an off-chain MongoDB database to support seat management, scanner authentication, and notification delivery. The database is organized into five main collections as shown in Figure 4.4.

doi:10.6342/NTU202600625

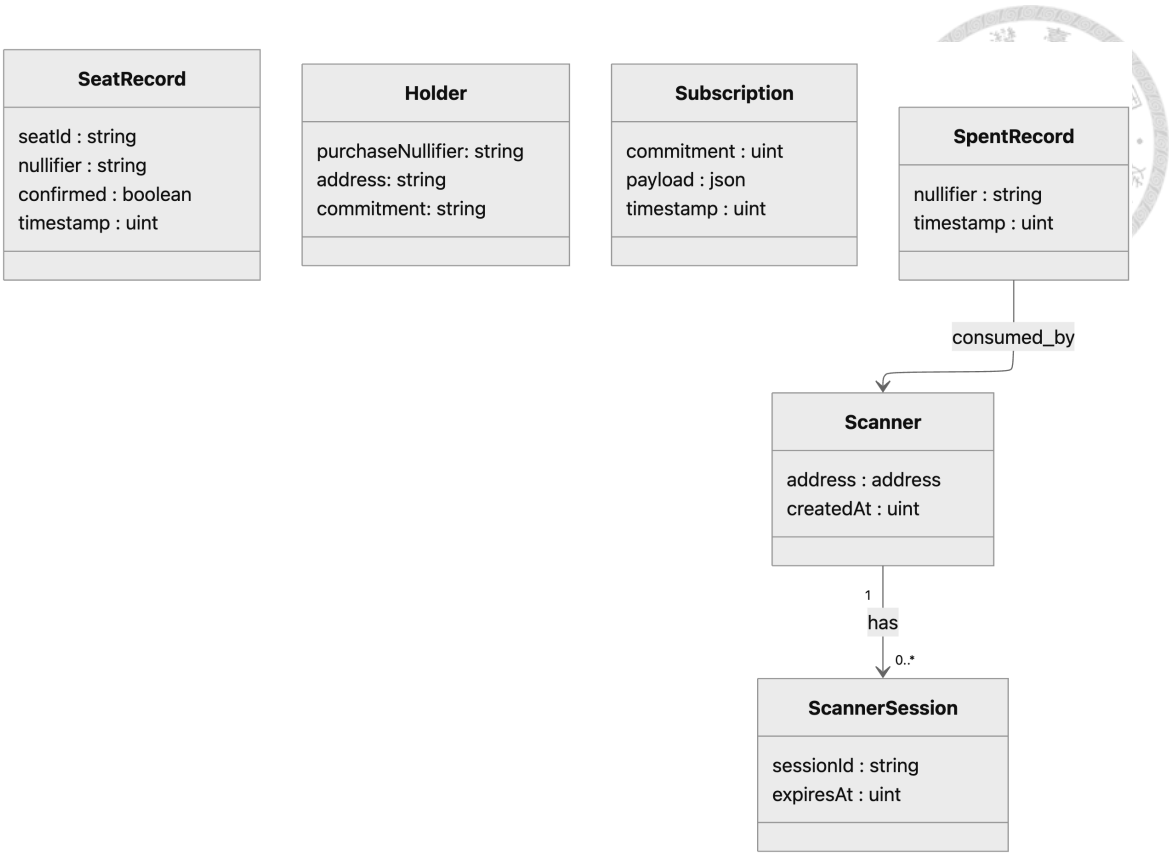


Figure 4.4: Off-chain database schema

4.3 Frontend Client

The frontend client serves as a thin orchestration layer that guides users through the ticket lifecycle while preserving privacy and enforcing correct interaction order. Rather than implementing trust-critical logic, the frontend visualizes system state and coordinates cryptographic operations, contract interactions, and user actions.

4.3.1 Purchase and Queue Flows

The frontend provides two primary acquisition flows: direct purchase and queue-based purchase.

As shown in Figure 4.5, in the purchase flow, users proceed through the following

doi:10.6342/NTU202600625

steps:

1. Connect an Ethereum wallet,
2. Authenticate on the device using a passkey and derive a device-bound identity,
3. Generate a World ID proof of personhood,
4. Select a seat,
5. Submit payment,
6. Access the ticket page.

In contrast, the queue-based flow omits seat selection prior to payment as shown in Figure 4.6. Users complete wallet connection, passkey authentication, and World ID verification before submitting payment and entering the queue. Seat assignment is deferred and finalized later by the system. This separation prevents linkage between user preferences and wallet addresses during high-demand events.

The frontend enforces the correct ordering of these steps and prevents users from triggering invalid actions (e.g., seat selection without a valid ticket). However, all policy enforcement and verification are performed by smart contracts or backend services.

4.3.2 Refund Flow

The refund interface allows ticket holders to reclaim escrowed funds in a privacy-preserving manner. Users reconnect their wallet and authenticate using the same device-bound passkey before submitting a refund claim. The frontend ensures that only the legit-

doi:10.6342/NTU202600625

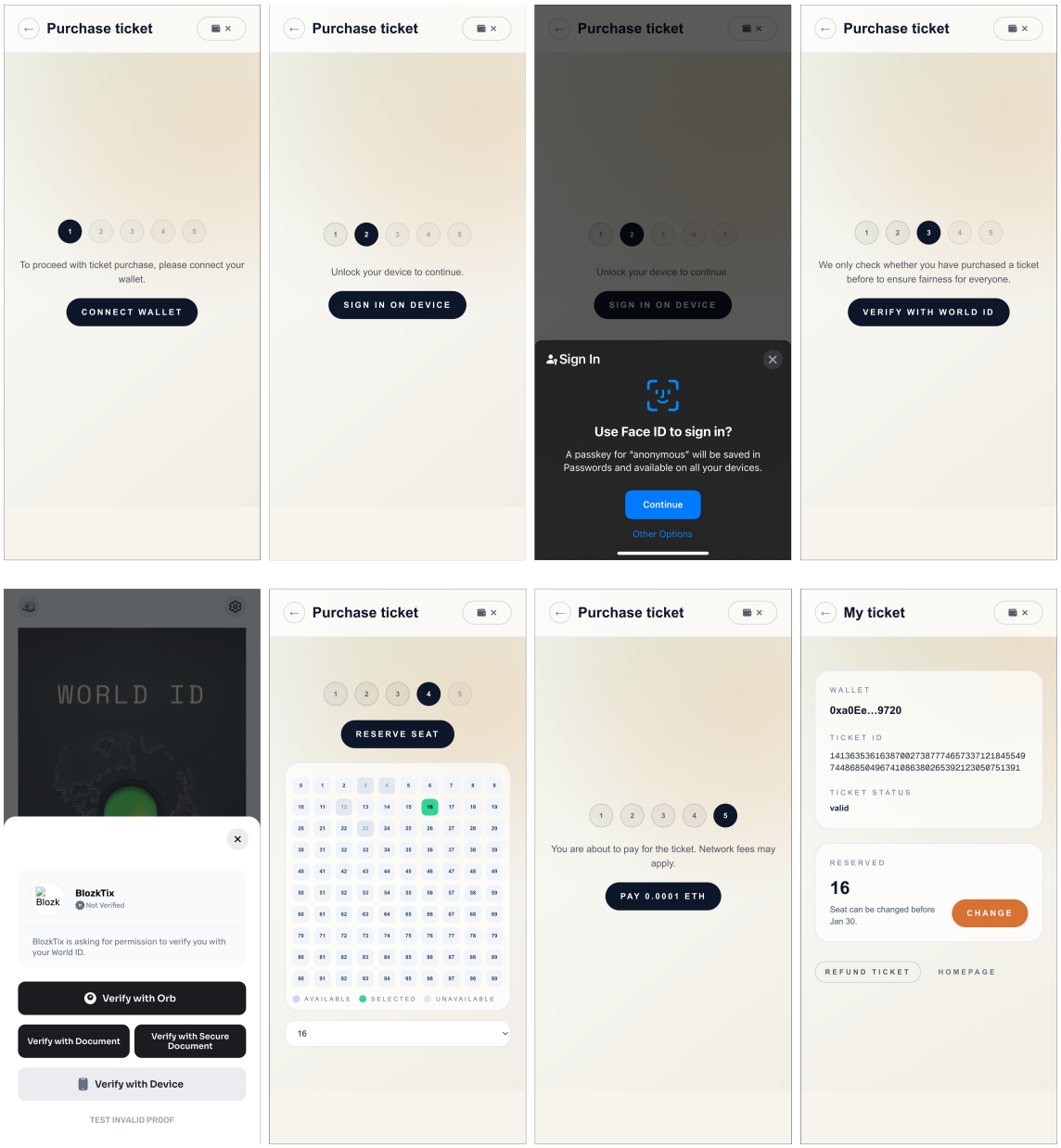


Figure 4.5: Purchase flow interface

doi:10.6342/NTU202600625

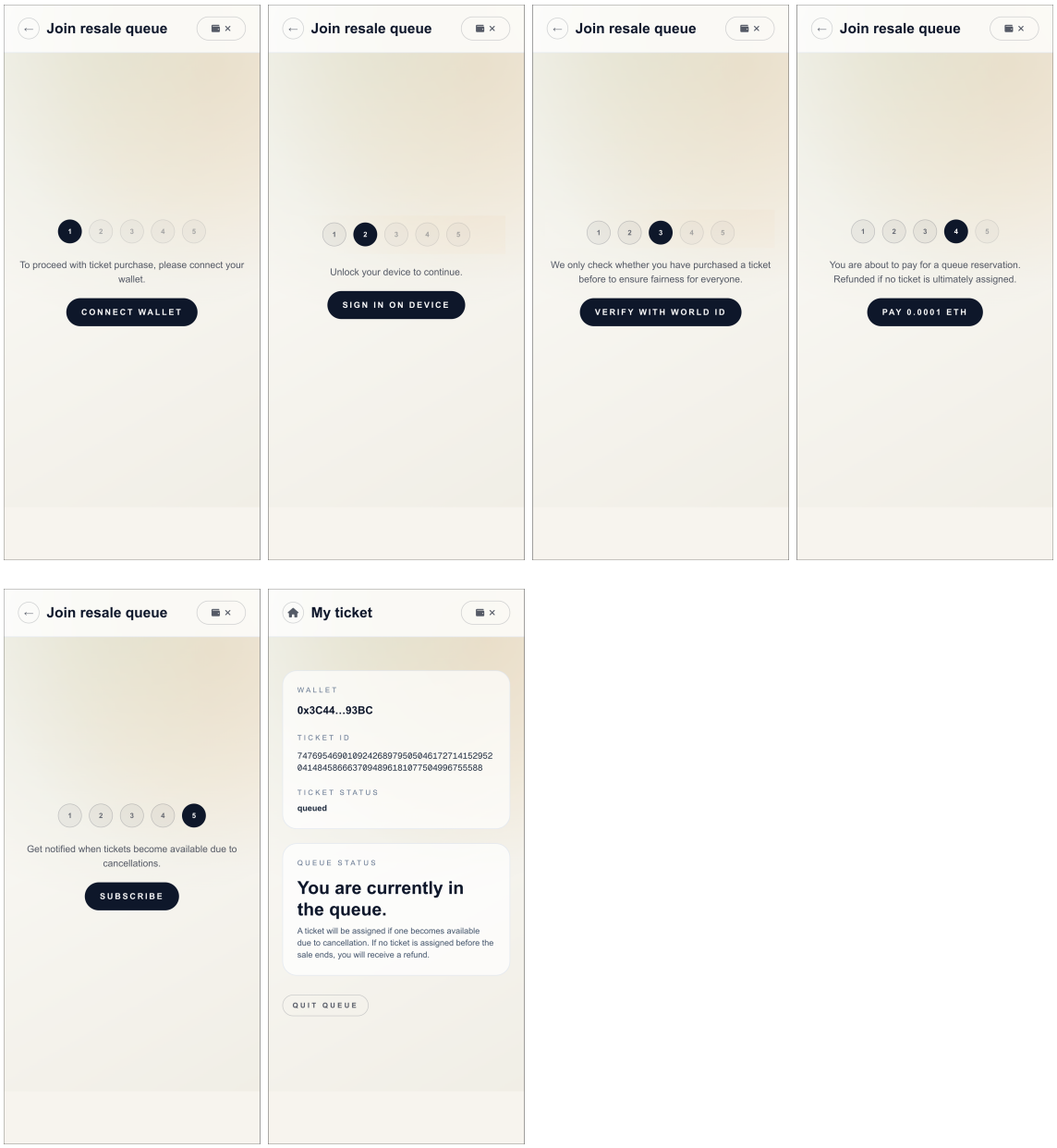


Figure 4.6: Queue flow interface

doi:10.6342/NTU202600625

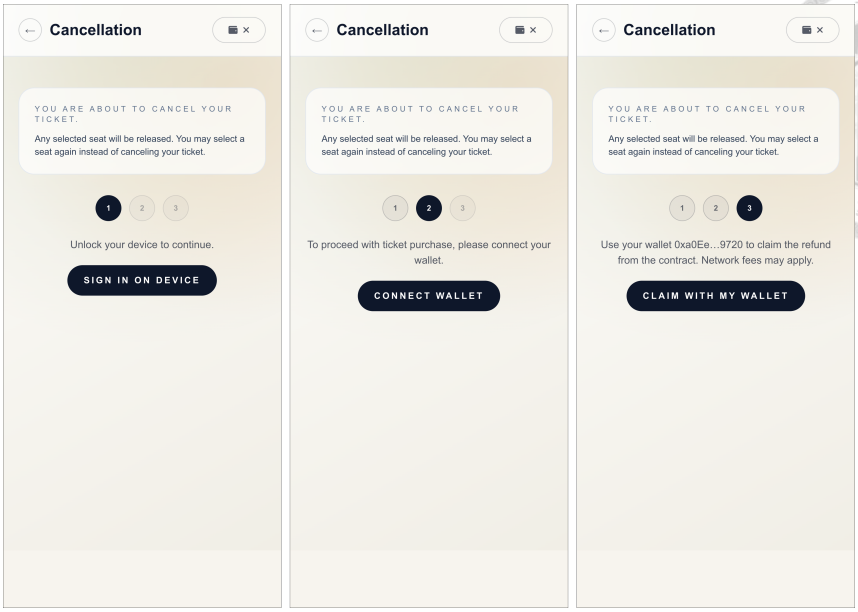


Figure 4.7: Refund interface

imate device can initiate the refund process, while the actual validation and fund transfer are executed on-chain. Figure 4.7 shows a screenshot of the refund interface.

4.3.3 Administrative Dashboard

The administrative dashboard is designed for event organizers and supports a limited set of privileged actions, including event configuration, finalization of seat and entry data, and escrow settlement. The frontend provides structured access to these operations without embedding authorization logic, which remains enforced by the smart contract and backend services. Figure 4.8 shows a screenshot of the administrative dashboard.

4.3.4 Scanner Interface and Offline Entry Verification

To support efficient and privacy-preserving event entry, a dedicated scanner interface is provided as shown in Figure 4.9. Ticket holders generate zero-knowledge entry proofs on their own devices prior to arrival. These proofs are encoded as QR codes and verified

doi:10.6342/NTU202600625

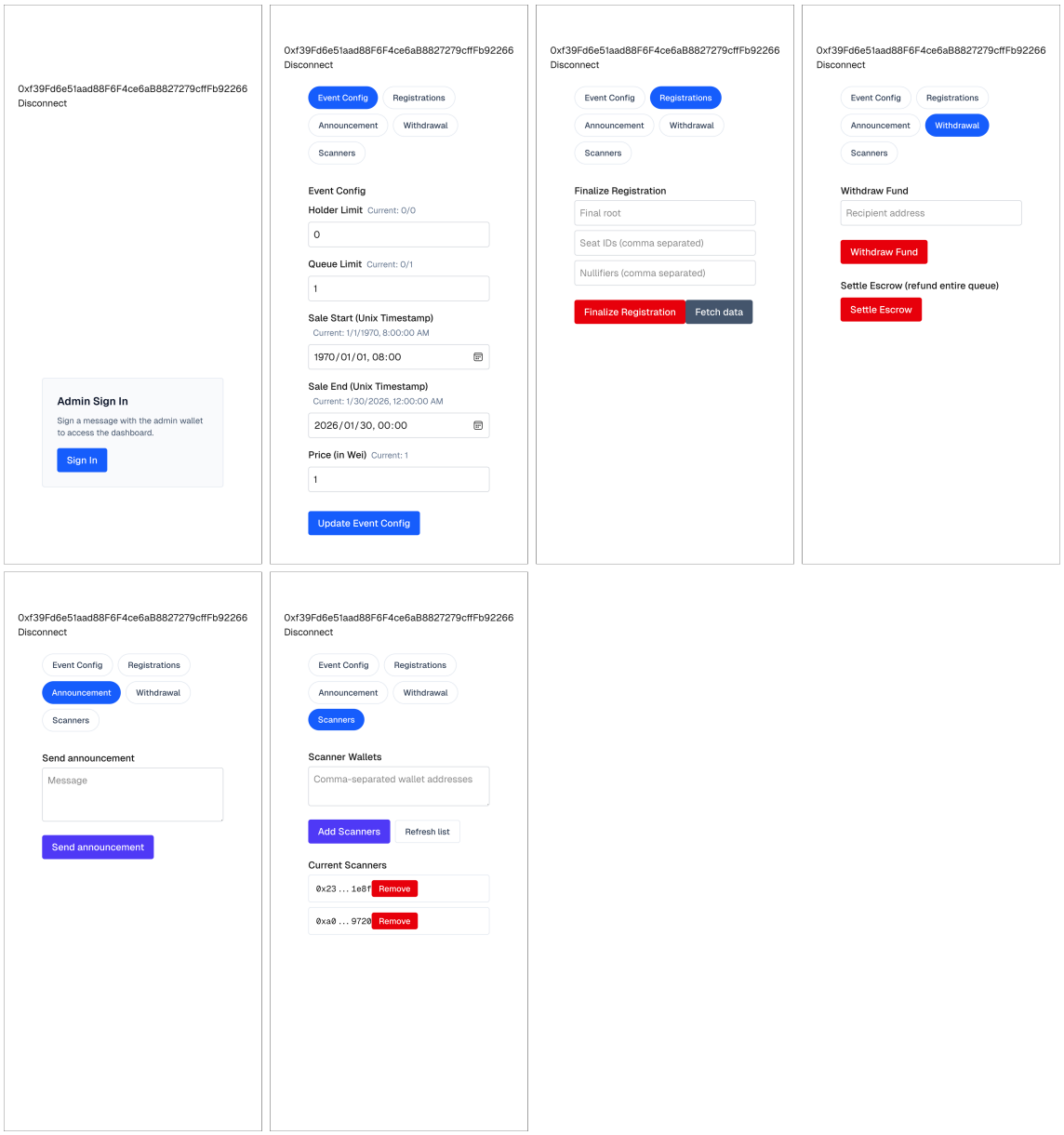


Figure 4.8: Admin dashboard

doi:10.6342/NTU202600625

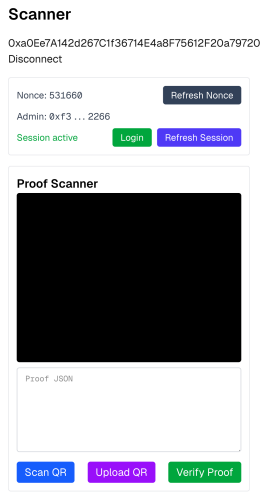


Figure 4.9: Scanner interface

locally by the scanner, enabling offline-capable validation with minimal latency.

The scanner interface does not require wallet connectivity or user login and only records spent nullifiers for later synchronization. This design ensures high throughput at entry points while avoiding real-time network dependencies.

4.3.5 Design Considerations

Several key design principles guide the frontend implementation:

- Bound identity: Passkeys are used to derive identities on the client, preventing identity independently transfer.
- Minimal data exposure: The frontend never displays or stores wallet–seat linkages or personal identifiers.
- Thin client architecture: The frontend coordinates flows and generates proofs but does not verify them or enforce allocation policies.

doi:10.6342/NTU202600625

- Offline usability: Entry verification remains functional without continuous network connectivity.
- Signature-based admin actions: Administrative operations require cryptographic signatures rather than login sessions, reducing attack surface.

Overall, the frontend client acts as a state-driven interface that enables secure, privacy-preserving interaction with the system while minimizing its role in trust-critical decisions.

doi:10.6342/NTU202600625

Chapter 5 System Evaluation

5.1 Analysis

This section analyzes the primary issues addressed by the proposed system, focusing on ticket scalping and user privacy leakage. The identified issues and corresponding mitigation mechanisms are summarized in the following tables.

5.1.1 Scalping-related Threats

Ticket scalping aims to unfairly acquire tickets or profit from resale by exploiting technical or social loopholes.

We assume that adversaries may control multiple wallets, automate transactions, or attempt off-chain attacks, but cannot break standard cryptographic primitives or compromise trusted hardware. The main scalping-related threats and their mitigations are summarized in Table 5.1.

doi:10.6342/NTU202600625

Table 5.1: Scalping-related Threats and Mitigations

Threat	**Description**	**Our Mitigation**
Automated Bulk Purchase	Using bots or scripts to rapidly buy large quantities of tickets upon release.	Proof of Personhood (PoP) limits its purchases to unique individuals, preventing mass acquisitions.
Multiple Account Creation	Creating numerous fake accounts to bypass purchase limits.	PoP ensures each account corresponds to a unique individual, restricting multiple accounts.
Off-chain Ticket Resale	Selling tickets outside the platform to avoid restrictions.	Smart contract – enforced non-transferability and refund mechanisms prevent unauthorized transfers.
Queue Manipulation	Exploiting queue systems to gain unfair priority in ticket allocation.	Transparent smart contract logic and fair queue-based promotion for refunds ensure equitable access.

As summarized in Table 5.2, blockchain-based approaches typically rely on wallet ownership, which allows tickets to be transferred through private key sharing, delegated signing, or custodial services with minimal friction. While some SSI- or ZKP-based systems introduce identity binding, they often lack mechanisms to prevent the practical transfer of ticket usage rights.

Our system adopts passkey-derived, event-scoped identities bound to proof of person-

doi:10.6342/NTU202600625

hood to mitigate unauthorized transfer. Although passkeys may be synchronized through passkey managers under explicit user consent, such transfer requires account-level access and user interaction, making it unsuitable for automated or large-scale resale. Compared to wallet-based designs, this approach significantly increases the cost and observability of off-chain ticket transfer, thereby discouraging economically viable unauthorized resale while remaining compliant with existing WebAuthn standards.

Table 5.2: Comparison of anti-scalping and security mechanisms across ticketing systems

Threats	**Traditional platforms**	**Blockchain ticketing**	**SSI / ZKP ticketing**	**Our system**
Sybil attack / bulk purchase	Real-name registration	On-chain identity or wallet-based binding	Identity binding	**Proof-of-personhood binding with on-chain enforcement**
Speculative profit	Not addressed	Price cap, royalty, or controlled resale	Not addressed	**Price-cap with queue-based resale**

Continued on next page

doi:10.6342/NTU202600625

Table 5.2 – continued from previous page

Threats	Traditional platforms	Blockchain ticketing	SSI / ZKP ticketing	Our system
Unauthorized transfer	Not addressed	Wallet-based ownership	Not addressed	**Passkey-derived event identity with high-friction transfer, bound to proof of personhood**
Replay attack	Time- or nonce-based request	Time- or nonce-based signature	Time- or nonce-based proof	**Nonce-based QR zero-knowledge proof**

5.1.2 Privacy-related Risks

In addition to fairness, preserving user privacy across different stages of the ticket lifecycle is a core design goal. Table 5.3 summarizes privacy-related risks and how the system prevents unwanted linkability.

doi:10.6342/NTU202600625

Table 5.3: Privacy-related Risks and Mitigations

Threat	**Privacy Risk**	**Our Mitigation**
Identity-wallet linkage	Real-world identity is linked to blockchain wallet addresses	Anonymous proof of personhood using World ID zero-knowledge proofs
Seat-wallet linkage	Seat selections are correlated with wallet addresses	Zero-knowledge seat selection proofs decouple seats from wallets
Entrance-wallet linkage	Physical presence at the venue is linked to wallet addresses	Zero-knowledge entrance proofs verify validity without revealing wallet identity

While the proposed mechanisms significantly reduce scalping incentives and prevent cross-stage linkability, the system does not attempt to eliminate all forms of abuse. Physical coercion, voluntary ticket sharing, and collusion between users and organizers are considered out of scope. Instead, the design focuses on raising the economic and technical barriers for large-scale scalping while preserving user privacy under realistic assumptions

doi:10.6342/NTU202600625

Table 5.4: Privacy risk comparison across ticketing systems

Risks	Traditional systems	Blockchain systems	SSI systems	ZKP systems	Our system
Personal data exposure	Real-name registration required	Pseudonymous (wallet-based)	Data minimized via credentials	Unlinkable and anonymous	**Anonymous proof of personhood**
Cross-event traceability	Traceable by platform	Publicly traceable on-chain	Limited, issuer-dependent	Unlinkable and anonymous	**Unlinkable and Anonymous**
Wallet linkage (seat / entry)	Wallet and seat/ entry linked by system	Wallet, seat, and entry publicly linkable	Data minimized but reusable	Unlinkable and anonymous	**Unlinkable and anonymous seat selection and entry**

Table 5.4 compares the privacy risks of the proposed system against traditional, blockchain-based, SSI-based, and other zero-knowledge proof–based ticketing systems.

Compared to traditional and blockchain-based ticketing systems, our approach achieves unlinkability across users' real-world identity, wallet, seat selection, and event entry, effectively eliminating cross-event and cross-phase traceability. While our design adopts the principles of identity separation and data minimization from SSI-based systems, we

doi:10.6342/NTU202600625

further address a critical limitation: even privacy-preserving SSI servers may still observe user participation and potentially correlate identities through issuer-side knowledge or collusion.

In contrast, our system relies on zero-knowledge proofs for both ticket purchase and seat selection, ensuring that neither the platform nor any off-chain service can associate a user's wallet with a specific seat or attendance record. Existing ZKP-based ticketing systems typically focus on anonymous purchase or entry verification, but do not address seat-level anonymity, which can expose users to physical tracking or targeted harassment in real-world venues. By extending zero-knowledge guarantees to seat selection and event entry, our system provides stronger protection for personal safety and on-site privacy, while preserving fairness and verifiability.

5.2 Performance Evaluation

This section evaluates the performance of the proposed system, focusing on the overhead introduced by group management and zero-knowledge proof operations. The evaluation aims to demonstrate that the system can be practically deployed for real-world events while preserving privacy and fairness guarantees.

All experiments were conducted on three representative device types: a consumer-grade laptop(MacBook Air M4, RAM 16GB), a tablet device(iPad Pro 4th Generation), and a server environment(in the laptop). Group sizes ranging from 1 to 5,000 members were evaluated to reflect small to medium-scale event scenarios. The time taken is averaged over 100 runs for each configuration to ensure statistical significance. However, the the result may vary due to background process and device state.

doi:10.6342/NTU202600625

Table 5.5 summarizes the average time taken for key operations, including group setup, membership proof generation, and proof verification.

Table 5.5: Performance Evaluation of Zero-Knowledge Operations

Device/Group Size	**1**	**10**	**100**	**1,000**	**5,000**
Group Construction Time (ms)					
Laptop	0.10	0.93	9.90	100.83	514.20
Tablet	0.00	1.67	15.00	137.33	707.67
Server	0.10	1.21	10.51	113.50	578.53
Joining Group Time (ms)					
Laptop	0.07	0.17	0.40	0.63	0.50
Tablet	0.00	0.33	1.00	0.67	0.67
Server	0.17	0.26	0.33	0.78	0.68
Proof Generation Time (ms)					
Laptop	139.93	192.87	264.92	225.73	327.20
Tablet	199.67	328.00	373.00	387.67	428.00
Server	251.26	174.18	493.47	218.54	665.07
Proof Verification Time (ms)					
Laptop	6.80	6.53	6.70	7.27	7.17
Tablet	14.33	18.33	14.00	11.00	11.33
Server	7.97	7.97	6.79	7.65	10.76

doi:10.6342/NTU202600625

5.2.1 Group Construction and Membership Update

A meekly tree group construction is required to generate and verify proof used in the Semaphore protocol. Table 5.5 reports the time required to construct a group for different group sizes.

The results show that group construction time increases approximately linearly depend on the group size. For a group of 5,000 members, construction completes within 514–708 ms across all tested devices. This behavior is expected, as group construction involves computing hash values for all Merkle tree nodes.

In contrast, joining an existing group introduces negligible overhead. As shown in Table 5.5, the time required to add a new member remains below 1 ms across all group sizes and devices. This indicates that membership updates can be efficiently handled without affecting system scalability.

5.2.2 Proof Generation

Zero-knowledge proof generation is performed on the client side for both seat selection and event entrance. Table 5.5 summarizes the proof generation time across different devices and group sizes.

The results show that proof generation time grow moderately with the group size. This increase is attributed to the deeper Merkle tree required to prove group membership. However, device performance is the dominant factor affecting proof generation latency.

On consumer-grade laptops, proof generation completes within 140–327 ms even for groups of 5,000 members. Tablet devices exhibit higher latency, ranging from approx-

doi:10.6342/NTU202600625

imately 200 to 430 ms, which remains acceptable for interactive use. Although server environments show greater variance, there are no proof generation in server environment in our system.

Overall, proof generation completes within sub-second latency across all evaluated scenarios, which is suitable for ticket purchase and entrance verification workflows.

5.2.3 Proof Verification

Proof verification is performed by the scanner or verifier during event entrance. Unlike proof generation, verification cost is largely independent of group size.

As shown in Table 5.5, proof verification time remains stable across all group sizes, typically within 7–11 ms on laptops and servers, and below 20 ms on tablet devices.

The consistently low verification latency enables real-time verification at event entrances without introducing congestion or delays.

5.2.4 Summary

The performance evaluation demonstrates that the proposed system achieves acceptable performance across different group scale and device types.

- Group construction incurs linear overhead.
- Proof generation completes within sub-second latency on consumer devices.
- Proof verification remains consistently fast and independent of group size.

These results indicate that the system can support real-world event scenarios with

doi:10.6342/NTU202600625

thousands of participants while maintaining interactive performance and strong privacy guarantees.

5.3 Contract Gas Cost Evaluation

This section evaluates the gas costs associated with key smart contract functions in the proposed system. Gas costs directly impact transaction fees and overall system usability, making it essential to assess their practicality for real-world deployment. Table 5.6 summarizes the gas costs for primary contract functions across different event sizes, ranging from 1 to 1,000 participants.

The gas cost evaluation demonstrates that most user-facing functions exhibit near-constant gas consumption, regardless of the participants count in the system. Most of the operations show stable gas usage across different system sizes, ensuring predictable and fair transaction costs for individual users, which is important for large-scale events.

In contrast, batch-processing functions—notably `settleEscrow`—exhibit linear growth in gas consumption as the queue size increases and exceed block gas limits when the queue size approaches or exceeds approximately 500–1,000 participants, leading to

Table 5.6: Smart Contract Gas Cost Evaluation

Function	**Size 1**	**Size 10**	**Size 100**	**Size 500**	**Size 1,000**
configEvent	120635	120635	120635	120635	120635
purchase	71005	71005	71005	71005	71005
queue	156204	156204	156204	156204	156204
cancelQueue	51257	49119	49119	49119	54479
cancelTicket	69609	69585	69597	69597	76297
settleEscrow	50918	196948	1657252	8147492	reverted
setAuditCid	48817	48817	48817	48817	48817
withdrawFund	26254	26254	26254	26254	26254
addMember(Platform)	157857	153893	186880	249526	254714

doi:10.6342/NTU202600625

transaction reversion. As a result, batch settlement must be partitioned into multiple transactions.

Overall, the evaluation confirms that the system maintains constant-cost for most operations, and confines linear-cost processing to administrator-controlled batch refund. This design achieves a practical balance between scalability, cost predictability, and on-chain safety.

doi:10.6342/NTU202600625

Chapter 6 Discussion

6.1 Extensible Identity Verification via DID and Multiple Semaphore Groups

The proposed system currently adopts Proof of Personhood as the primary mechanism for identity deduplication; however, its design is not limited to a single verification source. The framework can be extended to support additional attribute-based verifications, such as age categories (child/adult/senior), nationality, or membership in specific organizations or communities.

At the implementation level, the platform contract can be extended to act as a trust broker for external Decentralized Identity (DID) systems and Verifiable Credentials (VCs). Through this mediated approach, users are able to locally aggregate multiple identity attestations while relying on the platform to enforce issuer acceptance policies. Different verification sources are mapped to independent Semaphore groups, allowing users to generate zero-knowledge proofs only for the required attributes without revealing their actual identities or credential contents. This approach mitigates issues of inconsistent trust assumptions and governance across heterogeneous VC issuers.

By design, the system adheres to the principles of data minimization and selective

doi:10.6342/NTU202600625

disclosure, while avoiding the forced binding of multiple identity attributes to a single global identifier. As a result, cross-context linkability is significantly reduced, even when users participate in multiple verification scenarios.

6.2 Multi-Event Support and Organizer-Defined Verification Policies

The proposed architecture naturally extends to multi-event scenarios, where each event organizer may independently deploy its own server and smart contracts while defining event-specific policies. Organizers are able to configure not only the required identity verification mechanisms and Semaphore group combinations, but also ticketing policies such as queue mechanisms, ticket types, and event structures.

Specifically, the system can support different queueing strategies (e.g., first-come-first-served, randomized queues, or priority queues), multiple ticket types (such as VIP, general admission, or discounted categories), and complex event configurations including multi-date events and multi-area venues. These policies are enforced at the contract and protocol level, allowing organizers to tailor ticket allocation and access control according to the characteristics of each event.

To reduce integration overhead, the platform can provide standardized SDKs, reusable components, and contract factories, enabling organizers to adopt the protocol without implementing the full verification and ticketing logic from scratch. Identity proofs can be reused across organizer-managed websites through iframe-based modules or redirect-based interactions, allowing verification to be performed across different domains while preserving identity isolation and privacy.

doi:10.6342/NTU202600625

This design enables organizers to flexibly customize event policies without relying on rigid, platform-wide rules. At the same time, since verification logic and ticketing workflows adhere to a unified protocol design, interoperability is preserved and the emergence of new centralized trust points is avoided.

6.3 Broader Applicability Beyond Ticketing Scenarios

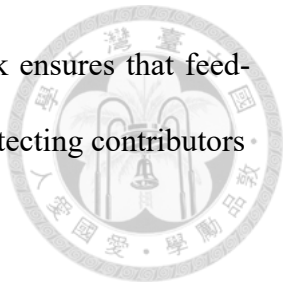
The identity and verification framework proposed in this work is not limited to ticketing systems. It can be applied in various scenarios that require uniqueness enforcement and eligibility verification without persistent identity disclosure. Public Transportation: Users may prove their eligibility to access transportation services (e.g., valid tickets or subscriptions) via zero-knowledge proofs, enabling entry and exit verification without revealing personal identities or travel histories.

- **Coupon Airdrop and Redemption:** The system ensures that each user can claim and redeem a coupon only once, preventing duplicate claims or secondary market abuse while avoiding linkage between coupon usage, payment records, and real-world identities.

- **Voting and Polling:** Group-based anonymous proofs enable one-person-one-vote guarantees, ensuring voter eligibility and non-reusability of votes while preserving the unlinkability of voting choices.

- **Online Exams and Certifications:** Candidate eligibility and single-attempt constraints can be enforced without permanently linking exam submissions or results to real-world identities.

doi:10.6342/NTU202600625



- **Anonymous Feedback and Whistleblowing:** The framework ensures that feedback originates from genuine and unique participants while protecting contributors from post-hoc identification or retaliation.

Ticketing systems represent a particularly challenging application domain, as they simultaneously require identity deduplication, fine-grained access control, resale constraints, and on-site verification under strict privacy requirements. Demonstrating the proposed framework in such a complex, end-to-end scenario illustrates its ability to handle realistic adversarial and usability constraints. The other applications discussed above can be viewed as simplified instances that reuse subsets of the same identity and verification mechanisms.

doi:10.6342/NTU202600625

Chapter 7 Conclusion

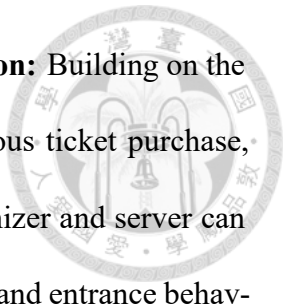
This thesis presents BlozkTix, a privacy-preserving and fairness-oriented blockchain-based ticketing system that addresses common issues in conventional ticket sales, including scalping, privacy leakage, and unfair resale practices. By integrating zero-knowledge proofs, the proposed system enables anonymous and unlinkable ticketing operation such as ticket purchase, seat selection, entry verification, and refund handling without relying on real-name identification. Blockchain smart contracts enforce ticketing rules, ensuring transparency and auditability while preventing unauthorized transfers and speculative profiteering. The system demonstrates how on-chain and off-chain components can be carefully coordinated to balance usability, security, cost efficiency, and privacy.

7.1 Contributions

The contributions of our work are summarized as follows:

- **Lifecycle-scoped anonymous identities:** We propose a lifecycle-scoped identity design in which different anonymous identities are used for different stages of the ticket lifecycle. At each stage, the user proves eligibility through zero-knowledge proofs, while the system cannot link actions across stages to the same individual.

doi:10.6342/NTU202600625



- **Anonymous purchase, seat selection and entrance verification:** Building on the lifecycle-scoped identity model, the system supports anonymous ticket purchase, seat selection, and event entrance verification. While the organizer and server can verify user eligibility using zero-knowledge proofs, seat choice and entrance behavior remain unlinkable to wallet activity and platform-level identity.

- **Passkey-derived ticket identity management:** To enhance security and usability, anonymous identities are derived from passkey authentication rather than being bound to wallet addresses. Identity material is not stored as a transferable secret and is re-derived when needed through device-mediated authentication, reducing the risk of off-chain identity transfer and mnemonic leakage or loss.

7.2 Limitations

Despite its advantages, the proposed system has several limitations:

- The proposed system requires users to possess both a blockchain wallet and a Proof-of-Personhood credential (e.g., World ID) and device support for Passkey, which may limit accessibility due to current adoption barriers.

- Blockchain transaction fees, confirmation latency, and client-side zero-knowledge proof computation introduce performance costs that may affect user experience and limit scalability for events.

- The current implementation assumes a trusted administrator for certain operations, such as signing refund permits, which introduces partial centralization.

doi:10.6342/NTU202600625

- Due to the intentional unlinkability between identities, the system cannot automatically associate recovered previously issued tickets if passkeys are lost.
- Advanced ticketing features such as multiple ticket types, multi-date events, configurable refund policies, and more resale mechanisms are not implemented in the current prototype, limiting its applicability across diverse event types.

7.3 Future Work

Several directions can be explored to further improve and extend the system to enabling more realistic event configurations and flexible ticket allocation policies:

- **Integration with other identity solutions:** To further enhance privacy and usability, the platform can integrate with off-chain decentralized identity solutions, allowing users to manage their identities and credentials more flexibly.
- **Scalability through grouped zero-knowledge membership:** To address scalability limitations, future work may explore partitioning participants into multiple zero-knowledge groups and applying more efficient proof verification to support larger-scale events while preserving anonymity and fairness guarantees.
- **Support for different ticket types:** The event system can be extended to support multiple seating areas and dates with different pricing models, such as VIP, general admission, and standing tickets, each with its own allocation and resale policies.
- **Support for different queueing mechanisms:** The ticket resale mechanism can be enhanced to support different queueing mechanisms, such as priority queues based

doi:10.6342/NTU202600625

on user reputation or loyalty programs, or lottery-based allocations for high-demand events.

doi:10.6342/NTU202600625

References

[1] Apple Inc. Introducing passkeys, 2023. https://developer.apple.com/passkeys/.

[2] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Succinct non-interactive zero knowledge for a von neumann architecture. In IEEE Symposium on Security and Privacy, 2014.

[3] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Succinct {Non-Interactive} zero knowledge for a von neumann architecture. In 23rd USENIX Security Symposium (USENIX Security 14), pages 781–796, 2014.

[4] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. Kroll, and E. Felten. Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. IEEE Symposium on Security and Privacy, 2015.

[5] V. Buterin. Ethereum: A next-generation smart contract and decentralized application platform, 2014. https://ethereum.org/en/whitepaper/.

[6] V. Buterin. Proof of personhood, 2021. https://vitalik.ca/general/2021/01/11/poap.html.

doi:10.6342/NTU202600625

[7] F. Casino, T. Dasaklis, and C. Patsakis. A systematic literature review of blockchain-based applications. Telematics and Informatics, 2019.

[8] S.-C. Cha, W.-C. Peng, T.-Y. Hsu, C.-L. Chang, and S.-W. Li. A blockchain-based privacy preserving ticketing service. In 2018 IEEE 7th Global Conference on Consumer Electronics (GCCE), pages 585–587. IEEE, 2018.

[9] Y. Chen and C. Bellavitis. Blockchain disruption and decentralized finance. Journal of Business Venturing Insights, 2022.

[10] P. Courty. Some economics of ticket resale. Journal of Economic Perspectives, 17(2):85–97, 2003.

[11] D. F. M. dos Santos. Smart e-tickets: buying authentic and trustworthy tickets with blockchain. Master's thesis, Universidade de Lisboa (Portugal), 2019.

[12] S. Eskandarian, S. Moosavi, and J. Clark. Sok: Transparent dishonesty: Front-running attacks on blockchain. In IEEE Security and Privacy Workshops, 2019.

[13] P. . S. Explorations. Semaphore documentation: How it works, 2023. https://docs.semaphore.pse.dev.

[14] S. Feulner, J. Sedlmeir, V. Schlatt, and N. Urbach. Exploring the use of self-sovereign identity for event ticketing systems. Electronic Markets, 32(3):1759–1777, 2022.

[15] FIDO Alliance. Passkeys: Passwordless authentication, 2022. https://fidoalliance.org/passkeys/.

[16] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. In Providing sound foundations for cryptography: On the work of shafi goldwasser and silvio micali, pages 203–225. 2019.

doi:10.6342/NTU202600625

[17] Google. Passkeys on google, 2023. https://developers.google.com/identity/passkeys.

[18] J. Groth. On the size of pairing-based non-interactive arguments. In Annual international conference on the theory and applications of cryptographic techniques, pages 305–326. Springer, 2016.

[19] K. Gurkan, W. J. Koh, and B. Whitehat. Community proposal: Semaphore: Zero-knowledge signaling on ethereum, 2020. Accessed July 1, 2021.

[20] M. Gysel, B. Ford, and L.-H. Merino. Blockchain-based Event Ticketing. PhD thesis, Master' s thesis: EPFL, 2023.

[21] J. Han, L. Chen, S. Schneider, H. Treharne, and S. Wesemeyer. Privacy-preserving electronic ticket scheme with attribute-based credentials. IEEE Transactions on Dependable and Secure Computing, 18(4):1836–1849, 2019.

[22] G. A. Haryadi, A. Zainudin, J.-M. Lee, and D.-S. Kim. Purenft: A blockchain-based ticketing system with lightweight ai for scalping prevention. In 2025 IEEE International Conference on Consumer Electronics (ICCE), pages 1–6. IEEE, 2025.

[23] P. Lafourcade, D. Mahmoud, G. Marcadet, and C. Olivier-Anclin. Transferable, auditable and anonymous ticketing protocol. In Proceedings of the 19th ACM Asia Conference on Computer and Communications Security, pages 1911–1927, 2024.

[24] X. Li, J. Niu, J. Gao, and Y. Han. Secure electronic ticketing system based on consortium blockchain. KSII Transactions on Internet and Information Systems (TIIS), 13(10):5219–5243, 2019.

doi:10.6342/NTU202600625

[25] M. L. Liu et al. A hybrid blockchain-based event ticketing system. PhD thesis, University of Saskatchewan, 2021.

[26] M. Ma and Z. Xie. Blockchain-powered ticketing ecosystem: A scalable framework for china's performance market. In 2025 International Conference on Culture-Oriented Science & Technology (CoST), pages 1–6. IEEE, 2025.

[27] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. https://bitcoin.org/bitcoin.pdf.

[28] S. Rafati Niya et al. Deti: A decentralized ticketing management platform. Journal of Network and Systems Management, 2022.

[29] F. Regner, N. Urbach, and A. Schweizer. Nfts in practice–non-fungible tokens as core component of a blockchain-based event ticketing application. 2019.

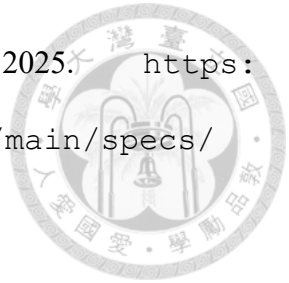
[30] B. Tackmann. Secure event tickets on a blockchain. In International Workshop on Data Privacy Management, pages 437–444. Springer, 2017.

[31] K. Thomas, F. Li, C. Grier, and V. Paxson. Protecting the internet from credential stuffing attacks. In Proceedings of the Internet Measurement Conference (IMC), 2017.

[32] U.S. Government Accountability Office. Event ticket sales: Market characteristics and consumer protection issues. Technical report, GAO, 2018.

[33] K. Verslype, B. De Decker, V. Naessens, G. Nigusse, J. Lapon, and P. Verhaeghe. A privacy-preserving ticketing system. In IFIP Annual Conference on Data and Applications Security and Privacy, pages 97–112. Springer, 2008.

doi:10.6342/NTU202600625



[34] vplasencia and oskarth. Semaphore v4 specification, 2025. https://github.com/privacy-ethereum/zkspecs/blob/main/specs/3/README.md.

[35] W3C Web Authentication Working Group. Web authentication: An api for accessing public key credentials, 2019. https://www.w3.org/TR/webauthn/.

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

[37] G. Wood. Ethereum: A secure decentralised generalised transaction ledger. Ethereum Yellow Paper, 2014.

[38] Worldcoin Foundation. World id: A privacy-preserving proof of personhood, 2023. https://worldcoin.org/world-id.

[39] X. Xu, I. Weber, and M. Staples. Architecture for blockchain applications. Springer, 2019.

[40] Y. YuanJiang and J. T. Zhou. Ticketing system based on nft. In 2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP), pages 01–05. IEEE, 2022.

[41] Y. Zhan, F. Yuan, R. Shi, G. Shi, and C. Dong. Pritkt: a blockchain-enhanced privacy-preserving electronic ticket system for iot devices. Sensors, 24(2):496, 2024.

[42] J. Zhao, S. Fan, and J. Yan. Blockchain-based ticketing systems: A survey. IEEE Access, 9:43742–43760, 2021.

[43] zk kit. Leanimt: An optimized incremental merkle tree, 2024. https:

doi:10.6342/NTU202600625

//github.com/zk-kit/zk-kit/blob/main/papers/leanimt/paper/leanimt-paper.pdf.

doi:10.6342/NTU202600625

Appendix A — Source Code

The complete source code for the BlozkTix system, including smart contracts, client applications, and zero-knowledge proof circuits, is available in the following repository: `https://github.com/fjwntu/blozk-tix`.

doi:10.6342/NTU202600625