mathematics

MDPI

Article

A Blockchain-Based Security Model for Aquatic Product Transactions Based on VRF-ZKP and Dynamic Reputation

Luxi Yu, Ming Chen, Yibo Zou, Yan Ge and Wenjuan Wang *

Key Laboratory of Fisheries Information, Ministry of Agriculture and Rural Affairs, College of Information Technology, Shanghai Ocean University, Hucheng Ring Road 999, Shanghai 201306, China; lucy_yuu@163.com (L.Y.); mchen@shou.edu.cn (M.C.); ybzou@shou.edu.cn (Y.Z.); yange_sh@163.com (Y.G.)
* Correspondence: wangwj@shou.edu.cn

Abstract

With the rapid development of online aquatic product trading, traditional centralized platforms are facing increasing pressure in terms of data security, privacy protection, and trust. Problems such as tampering with transaction records, weak identity authentication, privacy leakage, and the difficulty of balancing matching efficiency with security limit the further development of these platforms. To address these issues, this paper proposes a blockchain-based identity authentication and access control scheme for online aquatic product trading. The scheme first introduces a dual authentication mechanism that combines a verifiable random function with a Schnorr-based zero-knowledge proof, providing strong decentralized identity verification and resistance to replay attacks. It then designs a dynamic access control strategy based on a multi-dimensional reputation model, which converts user behavior, attributes, and historical transaction performance into a comprehensive trust score used to determine fine-grained access rights. In addition, an AES-PEKS hybrid encryption method is employed to support encrypted keyword search and order matching while protecting the confidentiality of order data. This paper implements a multi-channel architecture for aquatic product trading prototype system on Hyperledger Fabric. This system separates registration, order processing, and reputation management into different channels to improve concurrency and enhance privacy protection. Security analysis shows that the proposed solution effectively defends against replay attacks, key leaks, data tampering, and privacy theft. Performance evaluation further demonstrates that, compared to a single-chain architecture, the multi-channel design, while increasing security mechanisms, maintains a stable throughput of approximately 223 tx/s even when concurrency reaches 600–800 tx/s, ensuring normal operation of the trading system. These results indicate that this solution provides a practical technical approach and system-level reference for building secure, reliable, and efficient online aquatic product trading platforms.

Keywords: aquatic product trading; blockchain; identity authentication; dynamic reputation; searchable encryption; VRF-ZKP

MSC: 94A60; 68M25

Check for updates

Academic Editor: Antanas Cenys

Received: 29 December 2025
Revised: 13 January 2026
Accepted: 16 January 2026
Published: 20 January 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

With the rapid development of aquaculture and the cold-chain logistics system in China, demand for online transactions of aquatic products has grown rapidly. This has led to the emergence of numerous electronic trading platforms for both wholesale and retail [1].

However, most existing aquatic product trading platforms still rely on centralized databases for data storage and order management, which has exposed many problems in practice [2]. On the one hand, transaction data stored on a central server can be easily tampered with or forged, leading to trust risks such as opaque pricing, difficulty in tracing contract performance, and disputes over responsibility. On the other hand, the aquatic product supply chain is complex and multi-tiered, involves many heterogeneous participants, and generates diverse data formats, making it difficult for traditional systems to balance matching efficiency, order coordination, and security.

In recent years, blockchain technology has offered new solutions for the supply chain management of agricultural and aquatic products through its decentralized, traceable, and tamper-proof properties. Immutable ledgers and timestamps can provide verifiable traceability for origin information, inspection records, and cold chain temperature data [3]. Distributed consensus and decentralized storage reduce reliance on a single trusted party and improve system robustness. Smart contracts can automatically execute matching, settlement, and penalty rules on-chain, thereby reducing human intervention and trust costs. In addition, combining on-chain/off-chain hybrid storage and tokenization strategies can support data auditability and value transfer while maintaining system performance [4]. Blockchain also facilitates cross-organizational data sharing and, when integrated with IoT sensors, can support real-time on-chain recording of cold chain monitoring data and anomaly tracking, which improves transparency, reduces disputes, and strengthens regulatory oversight [5]. Despite these advantages, existing electronic trading systems still have significant shortcomings in security and privacy protection. In particular, there is a lack of systematic design for privacy protection, dynamic reputation assessment, and strong identity authentication during the transaction matching process [6,7]. It is therefore practically significant to design an on-chain aquatic product trading system that simultaneously supports strong identity authentication, fine-grained access control, and encrypted searchable matching.

At present, comprehensive studies that jointly consider identity authentication, access control, and searchable encryption in aquatic product trading scenarios remains limited [8]. Most platforms still rely on simple password-based authentication or escrowed public keys, which creates single points of trust and makes it difficult to resist credential theft [9,10]. Zero-Knowledge Proof (ZKP) have attracted wide attention in authentication scenarios because they can verify identity or attributes without disclosing sensitive information [11]. From early interactive proofs to non-interactive schemes based on the Fiat–Shamir heuristic, and then to efficient general-purpose constructions with practical implementations, state-in-art studies have shown that ZKP are now ready for real-world deployment [12]. Access control is another core component of information security, as it determines specific conditions access to certain resources. In aquatic product trading, access control not only distinguishes the basic permissions of buyers, sellers, and regulators, but can also be used to establish credit ratings based on participants' historical transaction behavior. These ratings can then be used to dynamically adjust matching priorities [13], thereby improving matching quality and overall platform satisfaction [14]. At the same time, directly publishing commercially sensitive order data on the blockchain creates serious privacy risks. To address this, researchers have introduced techniques such as searchable encryption and verifiable computation, which enable data retrieval and verification without revealing plaintext order information. Overall, these cryptographic and access control methods must be tailored and integrated according to the specific business requirements of aquatic product trading, so as to meet industry needs for privacy, regulatory compliance, and real-time matching.

Although these technologies have demonstrated superiority in other domains such as healthcare, finance and Internet of Vehicles, several key challenges remain in practical deployment for aquatic product trading [15]. First, ZKP-based authentication systems rely on high-quality random numbers; poor randomness can lead to proof forgery and undermine authentication security. Second, traditional access control mechanisms generally lack the ability to respond in real time to evolving attributes, behavior patterns, and reputation scores, making it difficult to achieve fine-grained dynamic authorization [16]. Third, cryptographic components, such as Verifiable Random Function (VRF), can produce proofs with substantial storage and verification overhead on-chain. Without compression or aggregation, these proofs increase on-chain costs and may introduce additional privacy risks. Consequently, there is a need for an integrated solution that addresses randomness risks, enhances the granularity of access control, and optimizes on-chain storage, while preserving strong cryptographic security [17]. For aquatic product trading scenarios, a comprehensive security solution must also prioritize the performance requirements of real-time cold-chain monitoring, batch-level privacy protection, and large-scale order matching, in order to achieve a practical, compliant, and efficient on-chain trading system.

In response to these challenges, this paper proposes a comprehensive blockchain-based solution for aquatic product trading that aims to improve transaction privacy protection, strengthen identity authentication, and optimize order matching efficiency. The main contributions of this study are as follows:

1. A VRF is introduced and combined with a zero-knowledge proof to achieve deterministic and verifiable random number generation, thereby mitigating risks associated with weak or reused randomness.
2. A multi-dimensional reputation evaluation model based on static roles is designed to support fine-grained access control and matching priority scheduling.
3. Advanced Encryption Standard and Public-key Encryption with Keyword Search (AES–PEKS) are integrated to perform symmetric encryption of order content and searchable encryption of keywords, enabling order matching over encrypted data.
4. A prototype system is implemented on Hyperledger Fabric, with Flask adopted as the back-end framework. By employing a multi-channel architecture, a complete functional loop—covering registration, matching, delivery, receipt, scoring, and reputation—is formed. Experiments are conducted to evaluate security and perform.

2. Literature Review

Scholars have conducted extensive research on the application of blockchain in aquatic and agricultural product supply chains. Zhang et al. proposed an improved ant colony algorithm to enhance matching efficiency and accuracy [18]. Zou et al. introduced a notary mechanism on top of a cross-chain model to alleviate data silos and achieve data interoperability across different ledgers [19]. Wang et al. adopted a multi-chain storage strategy to reduce the storage burden on a single chain and improve query efficiency [20]. Yan et al. designed a Zhanjiang aquatic product import–export platform that integrates a multi-layer architecture and collaborative mechanism, thereby optimizing traditional trade procedures and improving regulatory efficiency and product traceability [21]. In addition, significant progress has been made in research on food traceability, anti-counterfeiting, and cold-chain monitoring, providing strong technical support for food safety and traceability. Ge et al. proposed an enhanced HACCP program trusted and visible traceability model (EHCVTM) that combines semantic modeling with blockchain technology [22]. This model enables trusted, visible, and hierarchical management of quality traceability data in an aquatic-product HACCP system and improves the reliability and readability of traceability data. Zhang et al. designed a traceability system for frozen aquatic products by integrating

multi-sensor monitoring, smart contracts, and consensus mechanisms, thus achieving trustworthy on-chain data and tamper-resistant management from breeding to sales [10]. Liu et al. proposed a full-process, full-information food traceability method based on a trusted transmission protocol in a master–slave multi-chain blockchain [23]. Rossi et al. systematically integrated molecular analysis technologies with digital technologies and introduced a classification framework along two dimensions—technology maturity and information granularity—thereby providing a comprehensive technical roadmap and practical guidance for building efficient, transparent, and trustworthy food traceability systems [24].

In the context of data sharing and traceability, cross-domain identity authentication and fine-grained access control are crucial. They not only ensure the authenticity, security, and regulatory controllability of data at each stage of the aquatic product supply chain, but also directly affect the collaborative efficiency and overall credibility of blockchain-based traceability systems. To address these issues, Sun et al. used Schnorr signatures for identity authentication and Attribute-Based Access Control (ABAC) for attribute mapping, thus enabling dynamic access control [25]. Their scheme effectively addressed trust and security problems in cross-regulatory-domain data sharing and improved system throughput and response efficiency. Liu et al. combined ABAC with attribute-based searchable encryption to hide access control policies, achieving a necessary trade-off between functionality and privacy [26]. Sun et al. integrated a hybrid blockchain with a recommendation system to propose a food supply chain management method that improves supply chain efficiency and user experience through intelligent recommendations, while preserving data privacy and transparency [27]. In aquatic product trading, Teng [28] designed a credit evaluation mechanism for transaction matching models to optimize matching efficiency and enhance the satisfaction of both trading parties.

Overall, these studies have advanced transaction transparency and traceability in aquatic and agricultural product supply chains. However, systematic research and real-world deployment of strong identity authentication and dynamic access control specifically for aquatic product trading remain limited theoretically and practically. In particular, several key issues are still insufficiently addressed: (1) how to use cryptographic mechanisms to achieve decentralized identity authentication and effectively prevent credential theft and abuse; (2) how to implement fine-grained dynamic access control before order matching to enforce permission constraints, reduce information leakage, and lower computational overhead; and (3) how to apply searchable encryption to protect commercially sensitive keywords and order content during matching while minimizing the impact on matching efficiency.

To bridge these gaps, this paper proposes an end-to-end solution for blockchain-based aquatic product trading. During user registration and login, we introduce a joint identity authentication scheme based on a VRF and the Schnorr protocol, thereby enhancing credential security and resistance to theft while maintaining decentralization. During the order matching stage, by combining fine-grained dynamic access control with AES-PEKS based searchable encryption technology, permission filtering and privacy protection are achieved, thereby improving matching efficiency and overall security while safeguarding commercial secrets.

3. Design of Aquatic Product Trading Model

3.1. Design of Business Process of Online Aquatic Product Trading System

Online aquatic product trading mainly involves two types of entities: sellers (suppliers) and buyers (purchasers). Sellers publish orders with attributes such as species and quantity, and buyers submit matching requests through keyword-based searches. To ensure identity

traceability and prevent impersonation, the system uses a zero-knowledge proof based on the Schnorr signature scheme to provide strong authentication during registration and login. This method allows users to prove their identity without revealing their private keys [29]. In order to maintain verifiable matching capabilities while protecting commercial privacy, the order subject uses symmetric authentication encryption for encrypted storage, and generates public key searchable ciphertext for the search keywords to form an AES-PEKS search process [30]. To improve matching quality and suppress malicious behavior, the system introduces a multi-dimensional reputation model. Only entities whose reputation scores exceed a given threshold are granted higher matching priority and broader access permissions. Finally, considering the amount of on-chain data and consensus costs, the system adopts a multi-channel hierarchical storage architecture: large volumes of raw data that need to be kept confidential are placed in dedicated channels or off-chain storage to reduce the pressure of single-chain storage and consensus while taking into account audit traceability [31].

As shown in Figure 1, the business process of online aquatic product trading in the proposed system consists of five main stages: (1) registration and login, (2) order posting, (3) transaction matching, (4) transaction execution and scoring, and (5) contract and message querying.

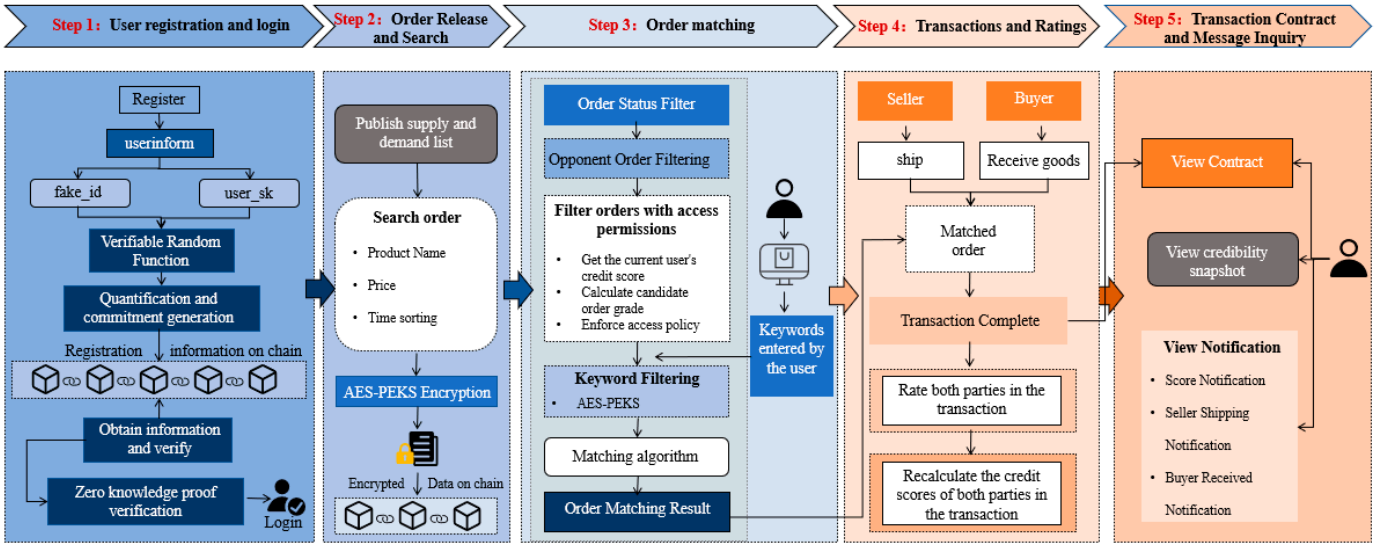


Figure 1. Business process of online aquatic product trading.

(1) **Decentralized registration and Schnorr ZKP-based identity authentication**. To ensure identity authenticity and privacy, and to support subsequent verifiability, users must register before using the trading system. After submitting identity information and passing local uniqueness checks, the system generates a VRF output and constructs the corresponding proof and vector commitment. These are stored on the blockchain to guarantee the immutability of user credentials. During login, the user derives the parameters of a Schnorr zero-knowledge proof from the on-chain VRF output and the current session challenge, and submits the proof to the system. A smart contract verifies the consistency of the on-chain registration record, the VRF output, and the Schnorr proof. Upon successful verification, thereby completing identity authentication.

(2) **Order posting based on AES and PEKS**. Users who have passed identity authentication can post aquatic product trading orders. Suppliers specify product categories, quantities, and other parameters to publish supply orders, while buyers post de-

mand orders according to their needs. Successfully published supply and demand orders enter an "open" trading state. To protect order privacy, the system constructs a searchable encrypted index using AES–PEKS. Before order submission, keywords are encrypted under a public key and core order content is symmetrically encrypted with a session key. Users can view their own orders in plaintext, while all orders are stored on-chain in encrypted form. After receiving the encrypted data, the chaincode writes them to the order chain, forming an encrypted order pool that supports verifiable, searchable, and tamper-resistant transaction record management [32].

(3) **Trapdoor retrieval and reputation-constrained order matching**. When initiating a matching request, the requester inputs the desired keywords. A smart contract first filters the candidate order set according to order status and the role of the order owner. It then invokes the predefined access control policy and carries out searchable encrypted retrieval over the encrypted index to match the query keywords. The preliminary retrieval results are further processed by the matching algorithm. After matching, the system returns the results to the requester and records the necessary audit logs.

(4) **Transaction execution and scoring**. Successfully matched orders move into a "matched" state and proceed to delivery and receipt. After the supplier initiates shipment, the order state changes to "shipped," pending confirmation of receipt. Once the buyer confirms receipt, the chaincode is automatically triggered to update the transaction status to "completed" and record the corresponding transaction hash on-chain. After the transaction is completed, both parties submit ratings based on each other's performance.

(5) **Transaction contract and system message query**. After a successful transaction, both parties can query historical transaction contract information via smart contracts at any time. The system publishes event notifications for each transaction stage in a notification panel, which can be used by relevant parties as audit evidence. Users can also view snapshots of their personal credit status to understand changes in their reputation scores and the sources of those evaluations.

3.2. Blockchain-Based Identity Authentication, Order Matching, and Multi-Channel Access Control Model

Existing online trading systems for aquatic products still rely heavily on traditional perimeter protection and static permission management. As a result, they are vulnerable to false registration, unauthorized access, and data tampering, which harms the interests of both trading parties and undermines the platform's reputation. To address these issues, this paper adopts the "never trust, always verify" principle from zero-trust architecture and combines strong identity authentication, least-privilege access, and continuous trust evaluation to design a blockchain-based identity authentication and access control model. In this model, blockchain is used to achieve decentralized credential management and tamper-proof audit logging. An efficient Schnorr-based identity authentication scheme ensures that only verified users can access system resources, while access rights are dynamically adjusted based on real-time trust assessment. The system is composed of three main modules: an identity authentication module, an order matching module, and a multi-channel module. The identity authentication module is built on the Schnorr protocol; the order matching module relies on dynamic access control as the core guarantee; and the multi-channel architecture provides business isolation and auditability [33]. The overall framework is shown in Figure 2.

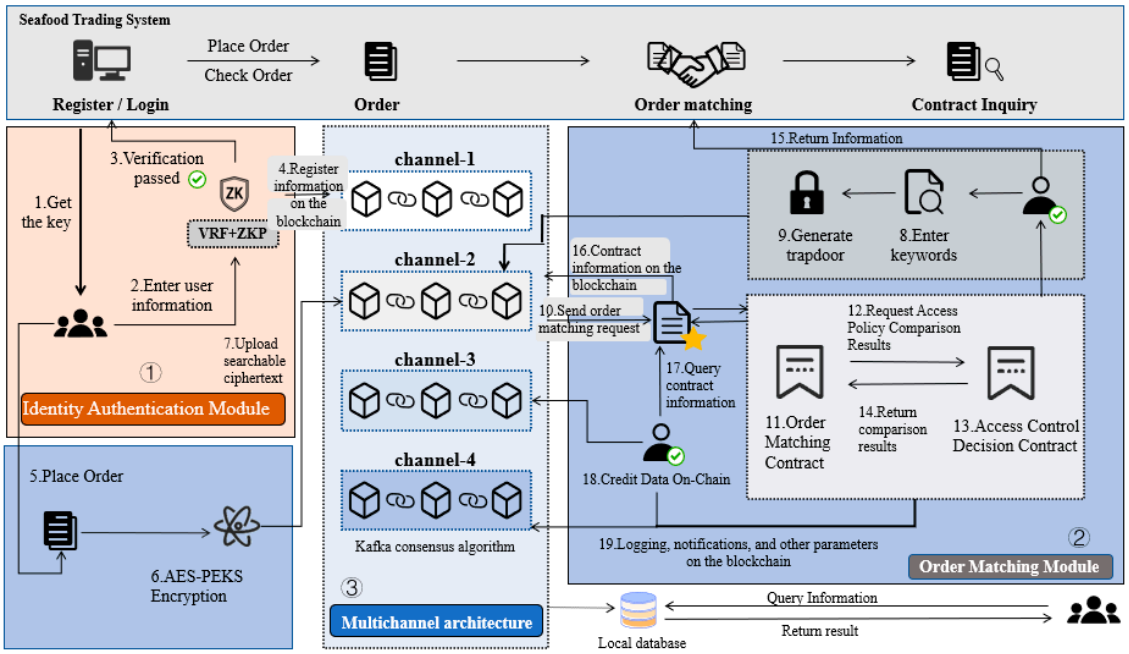


Figure 2. Overall structure of the proposed model. Document with an asterisk: The entire icon represents a smart contract; 2. User with a checkmark: The entire icon represents an authenticated user.

(1) **Identity authentication module.** The first module of this model is the identity authentication module, which is responsible for user registration and login functions. The system generates public–private key pairs for users and calculates the corresponding cryptographic proofs and user information vector commitments. These proofs and commitments are stored on the blockchain to ensure verifiability and tamper-proofness. Login authentication adopts a process combining VRF and ZKP: the user executes the VRF with private key and a unique identity identifier as input, generating a verifiable random output β and its proof vrf_{proof}; then β is combined with dynamically generated session challenges to generate a deterministic scalar r for Schnorr ZKP, instead of directly using β. The server should first verify the 'vrf_{proof}' to confirm the source and determinism of the random number. Then, based on the verified 'β', it should reconstruct the deterministic scalar '*r*' according to the same derivation rules, and finally verify the Schnorr signature to achieve dual authentication of "first-passive random number, then-passive identity," thereby effectively resisting replay and forgery attacks. To enhance robustness and evolvability, the system supports version updates of user information. When a user version is upgraded, the core VRF output remains unchanged; only the version identifier is updated and the vector commitment is recalculated to achieve a smooth and secure evolution of the state.

(2) **Order matching module.** The second module is the order matching module of the online aquatic product trading system. Its core design combines a trust-value-based evaluation model with searchable encryption to support keyword retrieval and order matching in an encrypted environment, while enforcing access control policies. The system first collects behavior indicators, attribute information, and historical transaction records for the target user. It then computes a comprehensive trust value from these multi-source features and dynamically updates the user's reputation level. Next, the reputation level is mapped to a corresponding data sensitivity level, and the system determines whether the user is permitted to access a specific order based on this mapping. If a registered user's reputation level is greater than or equal to the

access level required by a given order, the user is allowed to perform keyword search over the encrypted order index. When a user uploads an order, the session key is used to encrypt the order keywords, and the core order content is symmetrically encrypted. During retrieval, the querying user uses the private key to generate a trapdoor and matches it against the encrypted keywords. Once a match is found and the access policy is satisfied, the smart contract triggers the matching algorithm to finalize the order matching process.

(3) **Blockchain multi-channel network module.** The third module is the blockchain multi-channel network, designed to improve system scalability and privacy protection. The model adopts a multi-channel architecture in Hyperledger Fabric, where each channel maintains its own ledger state and serves specific business functions. The roles and corresponding storage contents of each channel are summarized in Table 1.

Table 1. Functional division of channels in the multi-channel network.

Channel Number	Functional Module	Storage Content	Access Organization
Channel-1	Registration and Authentication	VRF, ZKP, Vector Commitment Parameters	Org1 + Org2
Channel-2	Order Transaction	Encrypted Orders, Matching Records	Org1 + Org2
Channel-3	Reputation Evaluation and Aggregation	Scoring Data, Reputation Snapshot	Org1 + Org2
Channel-4	System Audit and Global Aggregation	Logs, Notifications, Global Parameters	System Management Node

By physically separating registration and identity data (Channel-1) from transaction data (Channel-2), the multi-channel network reduces the risk of sensitive information being inferred through correlation analysis. To maintain global consistency and system credibility, Channel-2 is designated solely for writing transaction data and reading reputation snapshots. Upon completion of each transaction and submission of a rating via Channel-2, Channel-3 immediately monitors for reputation update events, updates the user's reputation value in real time, and generates a new reputation snapshot to be written on the blockchain. Channel-4 supports cross-channel aggregation and parameter synchronization mechanisms, and retains cross-channel operation logs for subsequent auditing. Compared to traditional single-chain designs, the multi-channel architecture enables parallel processing of transactions, reduces ledger read/write conflicts, and significantly enhances both system auditability and privacy protection capabilities.

3.3. Identity Authentication Scheme Combining VRF and Schnorr Zero-Knowledge Proof

3.3.1. Cryptographic Foundation Based on Schnorr Protocol and VRF

The Schnorr zero-knowledge proof protocol allows a prover to demonstrate to a verifier that they possess the private key x without revealing it [34]. The core process is as follows: the prover selects a random number k, calculates the commitment $R = k \cdot G$, and combines this with the challenge $c = H(R||PublicInput)$ to calculate the response $s = k + c \cdot x$, ultimately generating the proof $\pi = (R, s)$. The verifier determines the validity of the proof by verifying $s \cdot G? = R + c \cdot P$.

However, the security of the traditional Schnorr protocol heavily relies on the unpredictability and uniqueness of the random number k. If k is leaked or reused, an attacker can recover the private key [35]. To address this issue and enhance the system's resistance to attacks, this scheme introduces an Elliptic Curve Verifiable Random Function (EC-VRF) to replace the traditional random source. VRF can deterministically generate a

pseudo-random output β based on a private key *sk* and an input *seed*, along with a publicly verifiable proof vrf_{proof}. Let the elliptic curve be defined over the finite field $GF(p)$, with base point G and prime order q. The steps for generating a verifiable random number using EC-VRF and its corresponding proof are as follows:

1. Generate a VRF key x based on the private key *sk*, and compute the corresponding public key as $pk = x \cdot G$.
2. Map the input message α to a curve point H_{point}, and compute the VRF hash point $\gamma = x \cdot H_{point}$.
3. Select a temporary random scalar k and compute the temporary public key point $U = k \cdot G, V = k \cdot H_{point}$.
4. Construct the challenge $c = Hash(H_{point}, U, pk, \gamma, V)$.
5. Calculate the response $s = k + c \cdot x \ (mod\ q)$.
6. Output (γ, π), where $\pi = (c, s)$.
7. Calculate the VRF output $\beta = H(hash_to_bytes(\gamma))$.
8. The verifier reconstructs U' and V', and computes c' to perform a consistency check by verifying that $c' = c$.

If the verification is successful, then γ is accepted as the VRF output for the input under the given public key. This method ensures the determinism and verifiability of random number generation without relying on external untrusted random sources [36]. In addition to random number generation, this scheme also focuses on the security of the seed in the VRF input. An adversary may attempt seed-grinding attacks by testing multiple seed values, precomputing the corresponding VRF outputs β, and selecting a β that is advantageous for manipulating subsequent protocol outcomes. To defend against such attacks, a two-way commitment protocol is employed to combine the user-supplied identity identifier with a server-generated random nonce to derive the seed. The resulting VRF output β is then bound to the session challenge and used as the input to the subsequent Schnorr protocol. This process makes it infeasible for adversaries to control the seed value or to precompute "favorable" outputs, thereby substantially increasing the practical cost associated with grinding attacks [37].

Although strong security is provided at the cryptographic level by this scheme, attackers can still launch side-channel attacks by analyzing the leakage of physical information during system operation. When the responses of a Schnorr ZKP is generated, if a non-constant-time algorithm or conditional branching is employed, attackers may infer bit information of the private key x by measuring the proof generation time and combining it with the known challenge c, thereby introducing the risk of timing attacks. Even more threatening is power analysis attack: if elliptic curve scalar multiplication is performed using the "double-and-add" algorithm, the power consumption difference between "point addition" and "point doubling" operations can directly reveal the bit pattern of the random number r. Once r is leaked, combined with the publicly available tuple (R, s, c), the private key x can be directly calculated using the formula $x = (s - r) \cdot c^{-1} \ mod\ q$. To defend against such attacks, this scheme employs a large-number arithmetic library whose execution time is independent of the operands, a blinding technique to sever the correlation between power consumption traces and the true scalar, and a constant-time Montgomery ladder algorithm to eliminate power consumption differences. In aquatic product trading platforms, the above protective measures are critical for safeguarding the security of node private keys.

3.3.2. Identity Authentication Scheme

Building on the above primitives, we construct an identity authentication scheme that combines Schnorr ZKP with a VRF. During the registration phase, a Schnorr key

pair is generated by the client on an elliptic curve, and the seed is bound to the user's unique identifier. The unpredictability and immutability of the seed are demonstrated by producing a VRF proof. Subsequently, the vectorized registration information is serialized and recorded on the blockchain via a linear vector commitment scheme, thereby providing an immutable on-chain proof of the registration.

During login, the system verifies: (i) the consistency between the VRF proof and the reconstructed deterministic scalar, (ii) the validity of the Schnorr proof, and (iii) the consistency between the on-chain vector commitment and the locally recomputed commitment. Incorporating the VRF intermediate value γ into the commitment strengthens its uniqueness and anti-forgery properties. Embedding version information into the user identifier supports lightweight version updates, preventing replay of revoked or expired credentials. Moreover, because each user's seed remains constant throughout its lifetime, the distribution of VRF outputs cannot be influenced by adversaries through manipulation of the seed. Without exposing private keys, this design ensures identity authenticity and session non-repudiation, while enabling verifiable reproduction of randomness, seed grinding attack prevention and minimizing on-chain storage of sensitive data. The registration and login process is shown in Figure 3.

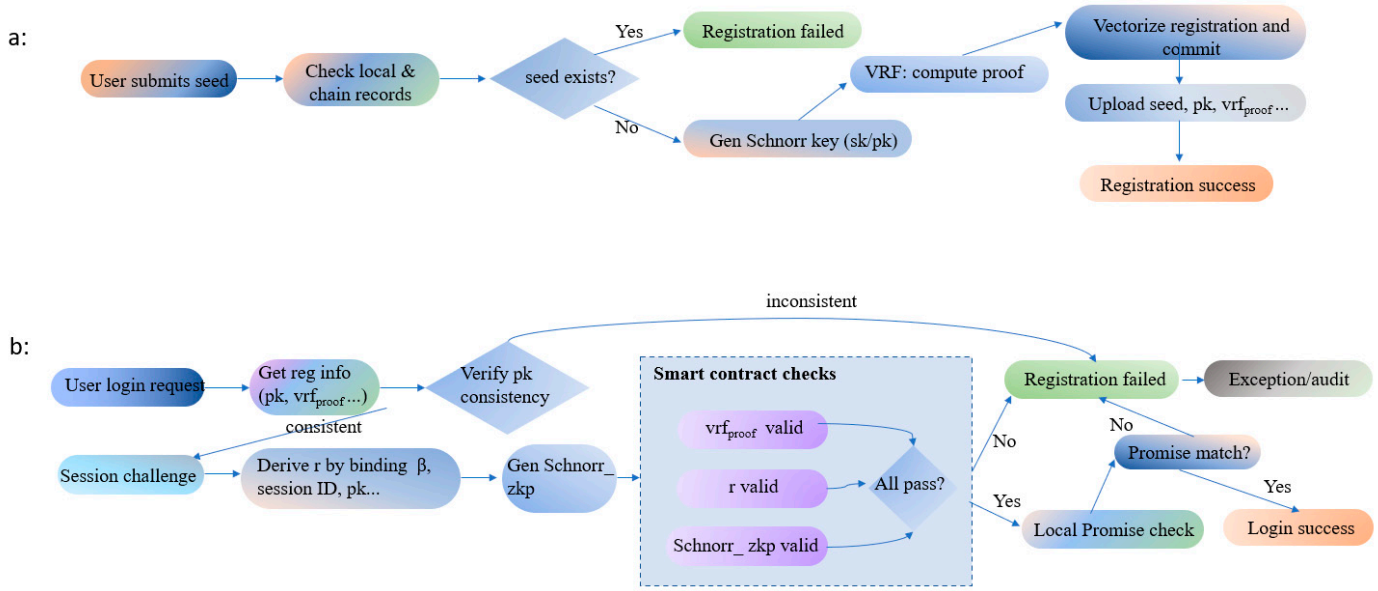


Figure 3. (**a**): User registration flowchart; (**b**): user login flowchart.

(1) **Registration phase:** As shown in Figure 3a, after a user submits their unique identifier on the front end, the system first queries this identifier in parallel on both the local database and the blockchain to prevent duplicate registrations. Only after confirming that no identical user exists will further processing continue. Subsequently, a unique seed is generated for the user by the system by combining the unique identifier submitted by the user with a system-generated random number, after which the seed commitment is solidified on the blockchain. Meanwhile, the client generates a Schnorr signature key pair (private key "*sk*" and public key "*pk*") based on the SECP256k1 elliptic curve. To reproduce and bind the user-specific pseudo-random value in a verifiable manner during subsequent login processes, the client uses a VRF with the user's private key and the seed as input, generating a VRF output pseudo-random value β and its proof "vrf_{proof}", thus ensuring that β can be verified on the blockchain. Then, the client maps the user registration information to a message vector and calculates a vector commitment, representing the registration data in a

compact and immutable form. Key information such as the user identifier, vector commitment, protocol version number, and public key are written on the blockchain, forming an immutable registration record.

(2) **Login Phase:** As displayed in Figure 3b, the server first retrieves the user's registration record on the blockchain and obtains the relevant materials stored during registration (including vector commitment, VRF proof, and public key). It then verifies the public key consistency, checking whether the public key submitted in the login request matches the on-chain record. The server generates a session key and sends it to the client. The client binds the parameter β, session ID, and public key according to the protocol, and derives a deterministic scalar r based on the bound value to serve as the random number in the Schnorr protocol. Subsequently, the Montgomery ladder algorithm is employed to generate a commitment R_s. The scalar r is then blinded, and constant-time modular arithmetic is utilized to generate a Schnorr ZKP, thereby demonstrating the correctness of the signature without exposing the private key. Next, the server calls the identity authentication smart contract to execute the zero-knowledge proof verification, including verifying the validity of the VRF proof and the correctness of the Schnorr signature. Simultaneously, it reconstructs the deterministic scalar r based on the on-chain record and the passed session parameters, and determines whether the reconstructed value is consistent with the scalar used in the signature. Finally, the server recalculates the message vector and its vector commitment of the registration information locally and compares the result with the on-chain commitment to confirm data integrity. If all the above checks pass, login verification is considered successful; otherwise, login is rejected, and corresponding exception handling and audit logging are triggered. The specific process is shown in Algorithms 1 and 2:

Algorithm 1 Generate composite proof (u: sk, seed; S: sessionID; B: challenge).

Require: u: sk, seed; S: sessionID; B: challenge
Ensure: schnorr_zkp

```
x ← sk
pk ← x × G
H_point ← ECVRF_Hash_to_curve(seed)
γ ← (x, H_point)
rho ← H_bytes(seed||b"rho")
U ← rho × G; V ← rho × H_point
c_vrf ← ECVRF_Hash_points(H_point, pk, γ, U, V)
s_vrf ← generateS(x, c_vrf, rho)
β← H_bytes(point_to_bytes(γ))
vrf_proof ← (H_point, γ, c_vrf, s_vrf)
challenge ← InitLogin(sessionID, seed)
r ← H_bytes(challenge, β)
R0, R1 ← G, G
for i from (bitlen(q) − 1) downto 0 do
    bit ← (r > >i) & 1
    (R0, R1) ← cswap(R0, R1, bit)
    R0 ← point_add(R0, R1)
    R1 ← point_double(R1)
    R0, R1) ← cswap(R0, R1, bit)
R_s ← R0
R_s_bytes ← point_to_bytes(R_s)
c_s ← H_bytes(seed||R_s_bytes) mod q
b ← RandomInZq()
x_blind ← (x + b) mod q
```

Algorithm 1 *Cont.*

25. t1 ← const_time_mod_mul(c_s, x_blind, q)
26. t2 ← const_time_mod_mul(c_s, b, q)
27. s_s ← const_time_mod_add(r, const_time_mod_sub(t1, t2, q), q)
28. schnorr_zkp ← (R_s_bytes, s_s, $\text{vrf}_{\text{proof}}$)

Algorithm 2 Verify composite proof (u: pk, seed; S: sessionID; B: challenge, schnorr_zkp).

Require: u: pk, seed; S: sessionID; B: challenge, schnorr_zkp
Ensure: accept/reject

1. (H_point, γ, c_vrf, s_vrf) ← $\text{vrf}_{\text{proof}}$
2. R_vrf_check ← s_vrf · G + (−c_vrf mod q) × pk
3. R_vrf_bytes′ ← point_to_bytes(R_vrf_check)
4. β_check ← H_bytes(point_to_bytes(γ)) mod q
5. P ← decode_pubkey(pk)
6. r_check ← derive_nonce_from_vrf(β, sessionID, seed, challenge)
7. R_s_check ← montgomery_ladder(r_check, G)
8. c_s′ ← H_bytes(seed | |R_s_bytes) mod q
9. R_check ← s_s · G + (−c_s′ mod q) × P

3.4. Reputation-Based Access Control and Searchable Encrypted Order Matching

3.4.1. Order Matching

The order matching process for registered users is illustrated in Figure 4. The system first filters orders with status "open" from the order pool. According to the current user's role (buyer or seller), it then selects candidate counterparty orders with the opposite role (i.e., buyer ↔ seller). Next, the system retrieves the current user's comprehensive reputation value and the data sensitivity level associated with each candidate order. Based on a predefined access control policy, it checks whether the user is authorized to access each order and removes all unauthorized orders from the candidate set. For the remaining authorized orders, the system applies searchable encryption to perform encrypted keyword matching, thereby completing retrieval while minimizing plaintext exposure. Finally, the matching algorithm generates the pairing results between authorized orders that satisfy both the access control policy and the keyword matching conditions. For first-time users, the system assigns a fixed initial reputation value (R0) (with the reputation range ([R0, 10])). The comprehensive reputation value is then dynamically updated according to the user's transaction behavior, attributes, and other relevant factors, and is subsequently used to drive access control decisions and matching priorities.

3.4.2. Reputation Assessment Model and Access Control

In traditional aquatic product trading prototype systems, access control typically relies on role-based access control or static, coarse-grained whitelist/blacklist strategies [38]. Permissions are fixed long-term after allocation, making it difficult to adapt to the dynamically changing risk and privacy requirements of trading scenarios. To address these issues, this study proposes introducing a dynamic reputation-based access control mechanism into the matching process. The scheme constructs a reputation model based on multi-dimensional user attributes. The system uses user behavior (transaction frequency, on-time rate, transaction amount, and transaction score), user attributes (real-name authentication, enterprise authentication, and registration duration, etc.), and historical trust performance (past assessed trust scores) as core influencing factors. These multi-dimensional inputs are mapped to a single comprehensive trust value through configurable weights to support subsequent access control and permission determination. Data sensitivity grading is implemented at the order level, for example, modeling sensitivity levels based on order price and marking

high-priced orders as highly sensitive. Before matching, a reputation threshold is used as the access determination condition; only when a user reaches this threshold can the corresponding order screening result be obtained.

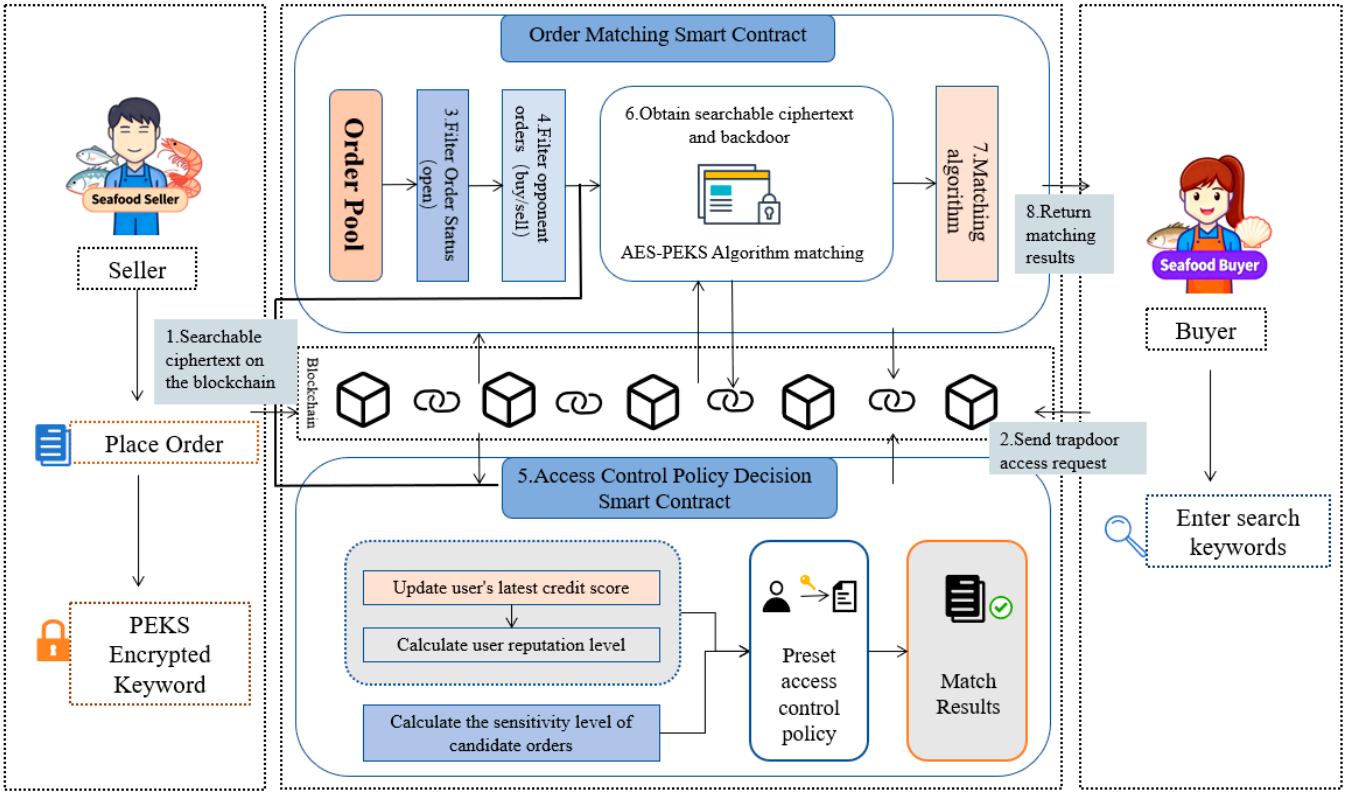


Figure 4. Order matching flowchart.

In the design of the reputation assessment model, consideration is given to the risks associated with coordinated attacks against the system. During the registration and login phases, robust VRF-ZKP authentication is employed to ensure that each blockchain identity uniquely corresponds to a single entity. Changes in reputation value are determined by both behavioral dimensions, such as transaction amounts, and attribute dimensions, which reflect differences in identity quality. This approach serves to mitigate Sybil attacks to some extent. Additionally, the system incorporates a simple anomaly detection mechanism: if user ratings consistently remain high or if multiple mutual ratings are observed between the same users within a short time frame, the influence of those ratings on the overall reputation score is automatically reduced. In this way, the risk of malicious collusion in rating behaviors is effectively curtailed.

Taking a registered user k as an example, let the evaluation period be T, and consider the update of its comprehensive trust value at evaluation node t. A weighted linear model is used, and calculation formula is shown in Equation (1):

$$R_k(t) = w_1 \cdot RH_k(t-T) + w_2 \cdot RV_k(t) + w_3 \cdot RA_k(T) \quad (1)$$

where $RH_k(t-T)$ is the historical reputation snapshot at the end of the previous evaluation period; $RV_k(t)$ is the behavioral trust value calculated based on the behavioral dimension within the current evaluation period T; $RA_k(T)$ is the attribute trust value calculated based on the attribute dimension within the current evaluation period. The coefficients w_1, w_2, and w_3 are configurable weights, satisfying $w_1 + w_2 + w_3 = 1$. To balance the continuity of reputation over time with sensitivity to recent behavior, the prototype system selected

$w_1 = 0.5$ (to improve stability), $w_2 = 0.3$ (to reflect recent behavior and risk characteristics), and $w_3 = 0.2$ (to capture static or semi-static trust cues). This weighted approach preserves the continuity of historical reputation while enabling the system to respond quickly to reputation changes caused by recent behavior.

Users' actual transaction behavior on the platform directly and dynamically reflects their ability to fulfill obligations and short-term risk characteristics; therefore, behavioral indicators are important signals for assessing short-term credibility and compliance risk. Therefore, this system selects four key behavioral factors within the period T and quantifies them:

(1) $Trade_{\text{time}}$: measured by the time interval between the buyer's confirmation of receipt and the seller's shipment (and the delay from the seller's successful matching to shipment). The smaller the delay, the timelier the performance [39];
(2) $Trade_{\text{frequency}}$: represented by the number of transactions within a unit period, reflecting the user's platform activity and participation;
(3) $Trade_{\text{money}}$: represented by the cumulative transaction amount within the period, reflecting the user's purchasing power and payment intention;
(4) $Trade_{\text{score}}$: as an interaction indicator between users, it can reflect the degree of recognition of the current user by other users. It is worth noting that the rating mechanism employs two-way signatures and timestamps, with each rating recorded on the blockchain. Only the two parties involved in a transaction are permitted to rate one another, thereby ensuring that rating behavior is strictly bound to genuine transactions. The behavioral trust value is calculated using a weighted linear form, as follows in Equation (2):

$$\mathrm{RV_k}(T) = w_1 \cdot Trade_{\text{time}} + w_2 \cdot Trade_{\text{frequency}} + w_3 \cdot Trade_{\text{money}} + w_4 \cdot Trade_{\text{score}} \quad (2)$$

where $w_1 + w_2 + w_3 + w_4 = 1$. The system selects $w_1 = 0.4$, $w_2 = 0.3$, $w_3 = 0.1$, $w_4 = 0.2$: Timeliness of transactions is given the highest weight to emphasize the criticality of fulfillment to the quality of aquatic product transactions; transaction frequency is secondary to capture activity and stickiness; transaction amount has a smaller weight to reduce the impact of occasional large orders; Transaction ratings are included in the calculation of Equation (2) only when the rater's reputation score exceeds 5, thereby ensuring that ratings from low-reputation users have limited impact, while those from long-term trustworthy users are assigned higher weight. The specific thresholds and weighting details for each factor are shown in Table 2.

Table 2. Behavioral scoring rules within a fixed period.

Behavior (Within a Fixed Period)	Scoring Rules
Transaction Timeliness	Interval < 12 h: 10 Interval 12 h–24 h: 6 Interval 24 h–48 h: 2 Interval > 48 h: 0
Transaction Frequency	>10: 10 5–9 times: 8 1–4 times: 2 0 times: 0
Transaction Amount	Cumulative amount > 2000: 10 Cumulative amount 1000–2000: 7 Cumulative amount 500–1000: 3 Cumulative amount < 500: 1
Transaction Rating	User rating: 1–10

For the reputation value $RA_k(T)$ of the attribute dimension, the system uses a weighted linear combination model for calculation, as follows in Equation (3):

$$RA_k(T) = \sum_{i=1}^{m} \beta_i \cdot f_i(A_i) \quad (3)$$

where A_i represents the i-th attribute, β_i is the corresponding influence weight, and satisfies $\sum_{i=1}^{m} \beta_i = 1$; To enhance the sensitivity to attribute differences and ensure the monotonicity of the mapping, the original attribute values are first normalized (e.g., mapped to [0, 1]) before fusion, and then the normalized values are amplified by the nonlinear transformation function $f(x) = \exp(x) - 1$ before participating in the weighted summation. If the four levels of an attribute are encoded as {0.2, 0.4, 0.6, 0.8}, then the corresponding conversion values are $f(0.2) = \exp(0.2) - 1$, $f(0.4) = \exp(0.4) - 1$, $f(0.6) = \exp(0.6) - 1$, $f(0.8) = \exp(0.8) - 1$. After weighting all attributes by a preset weight β_i, the attribute reputation value $RA_k(T)$ for that period is obtained.

Based on the above process, the system ultimately obtains the user's comprehensive reputation score R for the current period and maps continuous R values to discrete reputation levels (a higher R value indicates better reputation); the specific grading thresholds and mapping rules are shown in Table 3 below.

Table 3. Reputation level mapping.

Credit Scores (R)	Credit Rating
10~8.6	C5
8.5~7.0	C4
6.9~5.5	C3
5.4~4.0	C2
3.9~0.0	C1

When a registered user accesses a competitor's order, the system first obtains their latest overall trust value R, which serves as the primary basis for access determination. The system categorizes candidate orders into several price sensitivity levels based on their price attributes and sets a minimum reputation threshold for each level. Access is granted only when a user's overall trust value R is not lower than the threshold for the order's sensitivity level, as detailed in Table 4 below.

Table 4. Order price sensitivity levels.

Order_Price	Order_Level
>10,000	L5
10,000~5000	L4
4999~1000	L3
999~500	L2
Else	L1

In summary, users with higher trust scores are granted higher levels of access, enabling them to read and participate in order matching at a higher level of sensitivity, thereby effectively preventing low-reputation entities from intervening in large or high-risk transactions. To ensure the transparency and immutability of access decisions, the Access Policy (AP) is stored on-chain and automatically executed by smart contracts before matching.

To address scalability requirements posed by historical transaction volumes reaching thousands or even tens of thousands, a hierarchical aggregation and incremental update mechanism is incorporated into the system based on the original model. This approach

enables efficient and scalable reputation assessment in large-scale data scenarios. Transaction history is partitioned into discrete time windows according to a fixed period T, and within each window, behavioral summaries—including metrics such as transaction count, average score, and total transaction amount—are generated and stored in Channel-3, thereby reducing on-chain storage resource consumption. Following the completion of each new transaction, only the summary data for the current time window is updated, and incremental calculations are performed using the comprehensive reputation value from the previous calculation period, thus obviating the need for exhaustive recalculation of historical data. Additionally, reputation snapshots are periodically produced; during order matching, the access control module may directly retrieve the latest snapshot, avoiding latency and on-chain overhead associated with real-time calculations.

3.4.3. AES-PEKS Implementation of Searchable Encryption for Order Keywords

After the authorized order set is determined by the access control mechanism, users are allowed to perform encrypted keyword searches within this filtered subset. To balance data confidentiality with search functionality, this paper adopts a hybrid searchable encryption scheme based on AES-PEKS. In this scheme, sensitive fields in orders are encrypted using AES-GCM for symmetric authenticated encryption, ensuring both confidentiality and integrity of the stored data. At the same time, retrieval keywords are transformed into public-key searchable ciphertexts (PEKS), which serve as a publicly storable and matchable ciphertext index. Authorized users generate search tokens (trapdoors) using their private keys and specified query keywords. These tokens are then used to match against the stored PEKS ciphertexts, enabling keyword-based matching and retrieval directly over encrypted data, without decrypting the underlying plaintext orders. Thus, the system supports efficient keyword search while minimizing the exposure of commercially sensitive information. The overall searchable encryption process is illustrated in Figure 5.

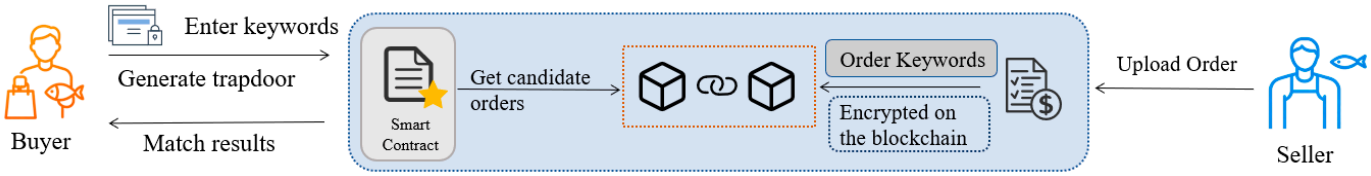


Figure 5. Flowchart of searchable encryption using AES–PEKS.

To defend against keyword guessing attacks, a random salt value bound to the order ID is incorporated as part of the input when generating keyword hashes. This measure ensures that identical keywords yield different hash values across distinct orders, thereby effectively preventing cross-order correlation analysis. In addition, keywords are padded with PKCS#7 before encryption to standardize the ciphertext length and eliminate the potential risk of leakage caused by length information. The detailed steps of the AES–PEKS-based searchable encryption and matching process are given in Algorithm 3.

Algorithm 3 AES-PEKS Encryption and Matching Test Algorithm (u: sk, keyword; S:Params; B: pk, orderID).

Require: u: sk, keyword; S: Params; B: pk, orderID
Ensure: ciphertext, trapdoor, matchResult
1. Salt ← H(orderID || timestamp || H(pk))
2. padded_kw ← PKCS7_Pad(keyword, 32)
3. kw_input ← padded_kw || salt
4. r ← GenerateRandomInZp()
5. H_w ← Params.HashToPoint(kw_input)
6. A ← Params.G.Mul(r)

Algorithm 3 *Cont.*

```
B ← H_w.Mul(r).Add(pk)
ciphertext ← (A, B, salt)
H_w' ← Params.HashToPoint(kw_input)
T_w ← sk × H_w'
trapdoor ← CreateTrapdoor(T_w, Hash(keyword))
left ← Params.Pairing(T_w, A)
H_w'' ← Params.HashToPoint(trapdoor.KeywordHash)
temp ← B.Sub(pk)
right ← Params.Pairing(H_w'', temp)
if left == right then
matchResult ← true
else
matchResult ← false
return ciphertext, trapdoor, matchResult
```

3.5. Smart Contracts

Based on the Hyperledger Fabric platform, this paper implements smart contracts in Go to build an identity authentication and access control system for online aquatic product trading. The system comprises three core smart contract modules: identity authentication, access control, and order matching. When a node receives a request, it invokes the corresponding chaincode to execute the required logic on-chain. In this way, the system supports efficient identity authentication, fine-grained and dynamic access control, and privacy-preserving order processing. The smart contract modules and their main functions are summarized as follows:

(1) Identity Authentication Smart Contract. Users generate registration information by inputting a unique identity identifier and upload it to the chain. When logging in, the system calls the identity authentication smart contract to dynamically generate session challenges and complete the verification of VRF proof, Schnorr proof, and the deterministic scalar r used for Schnorr. The corresponding logic code on the blockchain is as follows (Algorithm 4):

Algorithm 4 Identity Authentication Smart Contract (u: seed)

Require: u: seed
Ensure: AuthResult

```
RegisterUser(seed, commitment, ver, pihat, pk, vrfOutput, vrfProof)
InitLogin(sessionID, seed, timestamp)
require user[seed].isRegistered
challengeHex ← SHA256(sessionID, seed, timestamp)
sessionData ← {SessionID: sessionID, Seed: seed, Challenge: challengeHex, Status: "pending"}
return sessionID, sessionData
verifyZKP(sessionID, schnorrProof, vrfProof, vrfOutput, pk)
Session ← Getdata(sessionID)
if Session.Status ≠ "pending" then return reject
if not verifyVRFProof(vrfProof, vrfOutput, pk) then return "VRFfail"
if not verifySchnorrProof(schnorrProof, pk, Session.Challenge, vrfOutput, seed) then return "Schnorr fail"
Session.Status ← "verified"
return "accept"
verifyVRFProof(vrfProof, vrfOutput, pk)
return (vrfProof.s·G == vrfProof.R + vrfProof.c·pk) and (H_bytes(vrfProof.γ) == vrfOutput)
verifySchnorrProof(schnorrProof, pk, challenge, vrfOutput, seed)
```

Algorithm 4 *Cont.*

17. $r' \leftarrow$ reproduce_r(challenge, vrfOutput, seed)
18. if schnorrProof.r $\neq r'$ then return false
19. $C' \leftarrow$ generateC(seed, r')
20. return verifyEquation(schnorrProof.s, r', pk, c')

(2) Access Control Policy Decision Smart Contract. During the order matching process, the system calls the policy decision smart contract to implement access control. The contract maps user credit scores and order prices to corresponding credit levels and order sensitivity levels, and enforces access permissions to orders based on predefined access policies. Simultaneously, the contract maintains up-to-date credit snapshots and access logs to support subsequent auditing and monitoring. The following is a pseudocode description of the smart contract for this module (Algorithm 5):

Algorithm 5 Access Control Policy Decision Smart Contract (u: seed; D:orderID)

Require: u: seed; D:orderID
Ensure: AccessResult

1. Policies ← {
2. L1: {MinCreditScore: 0.0, MinCreditRating: C1},
3. L2: {MinCreditScore: 4.0, MinCreditRating: C2},
4. L3: {MinCreditScore: 5.5, MinCreditRating: C3},
5. L4: {MinCreditScore: 7.0, MinCreditRating: C4},
6. L5: {MinCreditScore: 8.6, MinCreditRating: C5}
7. }
8. UpdateCreditScore(seed, newScore, evaluationPeriod)
9. if userExists(seed) then
10. updateUser(seed, newScore)
11. CreditRating ← calculateCreditRating(newScore)
12. Snapshot ← CreditSnapshot(seed, newScore, CreditRating, timestamp, evaluationPeriod)
13. CheckAccess(seed, orderID)
14. user ← getuserReputation(seed)
15. order ← getOrder(orderID)
16. CreditRating ← calculateCreditRating(user.Score)
17. OrderLevel ← calculateSensitivityLevel(order.Price)
18. Policy ← getAccessPolicy(CreditRating, OrderLevel)
19. if CreditRating $\geq$ Policy.MinCreditRating and user.Score $\geq$ Policy.MinCreditScore then
20. AccessResult ← "granted"
21. else
22. AccessResult ← "denied"
23. accessLog ← {seed, orderID, user.Score, CreditRating, OrderLevel, AccessResult}
24. return AccessResult

(3) Order Matching Smart Contract. The order matching smart contract provides an automated, decentralized matching mechanism that balances data privacy and matching efficiency under a reputation-based access control framework. The following is a pseudocode description of the smart contract for this module (Algorithm 6):

Algorithm 6 Order Matching Smart Contract (u: seed; D:orderID)

Require: u: seed, searchKeyword; D:orderType
Ensure: matchResults

1. ExecuteOrderMatching(seed, searchKeyword, orderType)
2. openOrders ← filterOpenOrders(orderType)

Algorithm 6 *Cont.*

```
3.  oppositeOrders ← filterRoleMatchedOrders(openOrders, seed.Type)
4.  accessibleOrders ← CheckAccess(oppositeOrders, seed)
5.  keywordMatchOrders←applyKeywordMatching(accessibleOrders, searchKeyword)
6.  matchResults ← executeGreedyMatching(keywordMatchOrders, seed.Type)
7.  finalizeMatches(matchResults)
8.  return matchResults
9.  executeGreedyMatching(orders, userType)
10. if userType == "buyer" then
11.   sortedOrders ← sortByPrice(orders, ascending)
12. else
13.   sortedOrders ← sortByPrice(orders, descending)
14. return sortedOrders[0]
15. finalizeMatches(matches)
16. for each match in matches do
17.   UpdateOrderStatus(match.buyOrderID, "matched")
18.   UpdateOrderStatus(match.sellOrderID, "matched")
```

4. Results

4.1. System Design and Implementation

4.1.1. System Architecture

Based on the identity authentication and access control model for online aquatic product trading shown in Figure 2, the system is designed using a four-layer architecture: presentation layer, business logic layer, blockchain network layer, and data persistence layer. The overall architecture is shown in Figure 6.

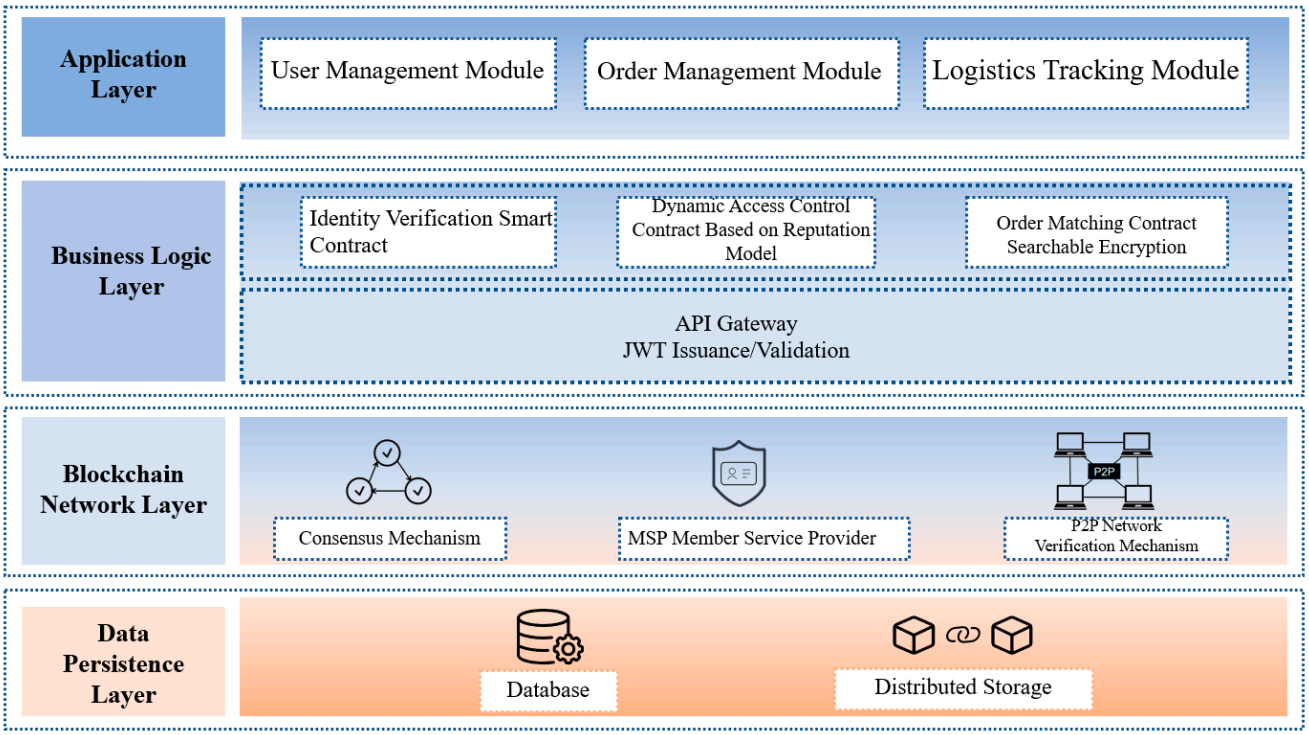


Figure 6. System Overall Architecture Diagram.

Application Layer: The application layer encompasses three core modules: user management, order management, and logistics tracking, achieving a closed-loop business process from user registration, order posting and matching, shipment/receipt confirmation to contract management and decision support. Each module interacts with backend services

through a unified interface, ensuring the continuity of business operations, security, and consistency of user experience.

Business Logic Layer: Responsible for receiving and processing user requests, executing business logic, and interacting with chaincode or smart contracts. User authentication and access control are primarily implemented through on-chain smart contract mechanisms and off-chain collaboration: Anonymous authentication employs a zero-knowledge proof method incorporating VRF to ensure that only verified entities can securely log in to the system without exposing their identity privacy; dynamic access control calculates a user's overall reputation value based on a reputation model, and access to an order is granted only if the user's reputation is not lower than the access threshold required by the target order. Searchable Encryption enables encrypted keyword retrieval during the order matching phase, completing matching and matching within the encrypted domain.

Blockchain Network Layer: The system adopts a multi-channel and multi-organization consortium blockchain architecture, containing two participating organizations (Org1 and Org2), representing the consortium nodes of the buyer and seller, respectively. Each channel independently maintains its ledger state to achieve transaction isolation, and member identity management and role-based access control are implemented through a Membership Service Provider (MSP). To ensure availability and consistency in a multi-node environment, the system employs the Raft consensus algorithm as the consensus protocol for the ordering service. Simultaneously, a Peer-to-Peer (P2P) network topology and transaction endorsement/verification mechanisms constitute the core of the network layer, used for inter-node communication, transaction propagation, and on-chain transaction verification and endorsement.

Data persistence layer: The layer consists of an on-chain ledger and an off-chain database. The on-chain ledger stores encrypted transaction summaries, hash indexes, and reputation snapshot summaries to provide tamper-proof credentials and audit trails. The off-chain database stores auxiliary data such as system operation logs, notification records, and plaintext contracts for efficient retrieval and auditing, thus achieving a dual-track storage strategy of "trustworthy on-chain and efficient off-chain." Users initiate transaction requests, which are processed by the Flask backend and invoked through the Hyperledger Fabric SDK to call the corresponding smart contracts. The smart contracts execute business logic and generate block data on the corresponding channels. Finally, the transaction results and reputation snapshots are written to on-chain and off-chain storage, respectively, to achieve the trustworthiness and traceability of aquatic product transactions.

4.1.2. Prototype System Implementation

This system is implemented on the Hyperledger Fabric consortium blockchain platform, with a MySQL database used for off-chain storage. The front end is developed in JavaScript using Vue 3 to provide an interactive user interface. The back end consists of two components: chaincode written in Go and business API services implemented with Flask (Python). The experimental environment is 64-bit Ubuntu 22.04.1 with Docker 18.09.7. The implementation focuses on on-chain identity authentication and reputation-based access control: the system supports multi-user registration and login and enforces order access and matching according to predefined reputation policies.

To illustrate system behavior, three representative users are considered: user01 and user02 act as sellers, and user03 acts as a buyer. After completing registration and login verification, user01 posts a supply order (variety: swimming crab; quantity: 10; price: 900) with a reputation level of C1. user02 posts another supply order (variety: swimming crab; quantity: 10; price: 499), also with a reputation level of C1. user03 submits a demand order (variety: swimming crab; quantity: 8; bid: 600), with a reputation level of C1. According to

the system's access policy, each reputation level is mapped to the set of order levels that can be accessed (e.g., C1 can only access L1-level orders). Based on the order price attribute, user01's supply order is classified as L2, whereas user02's supply order is classified as L1. Consequently, user03, with a C1 reputation level, can only access and attempt to match user02's L1 supply order.

Figure 7 illustrates the implementation of the order posting and order matching functions in the aquatic product trading prototype system. As shown in Figure 7a, user03, after successful identity verification and login, can publish a demand order through the "Publish Order" interface. As shown in Figure 7b, after submitting search keywords on the matching page, the system performs encrypted keyword matching and returns a suitable supply order; in this example, the matching result selects seller user02 as the counterparty.

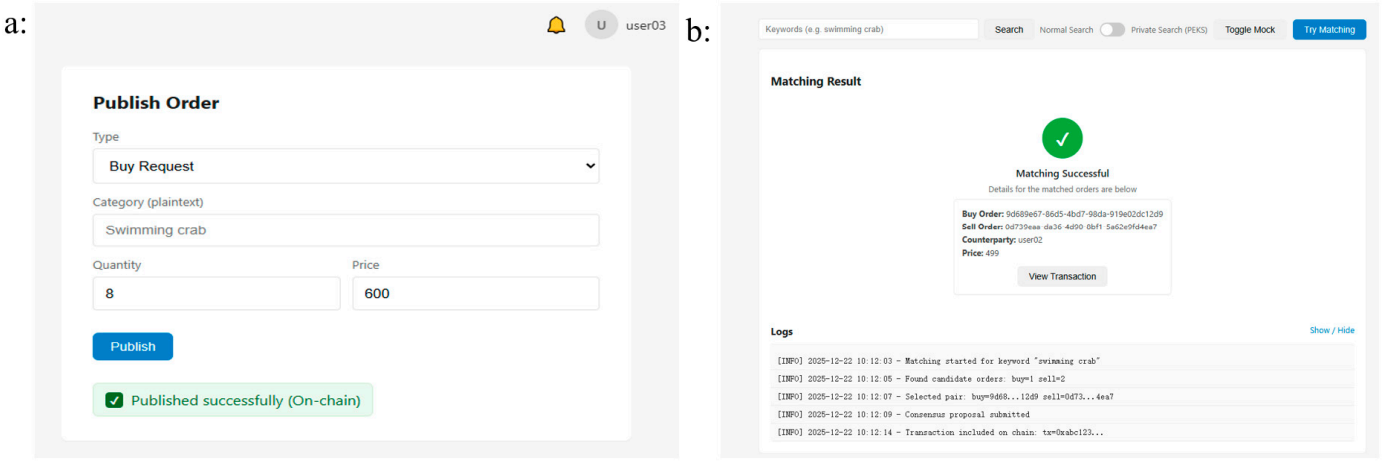


Figure 7. (**a**): Order release; (**b**): order matching.

Figure 8 illustrates the implementation of order rating and contract query functions in an aquatic product trading prototype system. Figure 8a shows that after an order transaction is completed, users can rate and evaluate the target order. Figure 8b shows that after the transaction is completed, both parties can view the contract information for the completed order.

4.2. Security Analysis

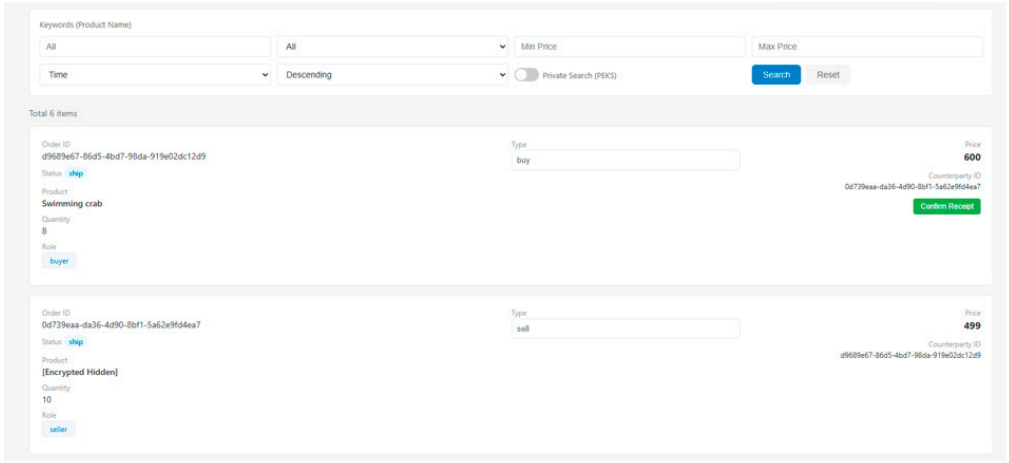
In the implementation, the proposed model targets common security problems in traditional aquatic product trading platforms, including data tampering, privacy leakage, identity forgery, and reputation fraud. Security mechanisms are deployed at multiple layers—encryption, data structures, and access control. To evaluate the effectiveness of the scheme, this section analyzes several representative attack types and their potential impact, and explains how the proposed design mitigates them.

4.2.1. Prevention of Random Number Replay Attacks

In the standard Schnorr zero-knowledge proof protocol, security critically depends on the unpredictability and one-time use of the commitment nonce (k) [40]. If (k) is predictable or reused across sessions, an attacker can exploit two signatures that share the same (k) and the corresponding public information to recover the private key (sk) via simple algebraic manipulation, leading to key compromise [41]. In addition, attackers can attempt replay attacks by resubmitting intercepted valid proofs. To eliminate these weaknesses, the proposed scheme integrates a verifiable random function (VRF) into the Schnorr protocol. The random parameter effectively used to derive the Schnorr nonce (and thus the commitment (R) is generated from the VRF output and is strongly bound to a session-specific seed. With-

out knowledge of the user's private key (sk), an attacker cannot forge a valid composite proof ($\pi = \left(R, s, vrf_{proof}\right)$). During login, the user submits only the challenge response and corresponding proofs. The verifier checks legitimacy by computing ($R' = s \cdot G - c \cdot P$) and comparing ($R'$) with the ($R$) included in the proof, while also verifying the VRF proof. Throughout this process, no sensitive information about (sk) is revealed, and the proof (π) satisfies the zero-knowledge property. Thus, the authentication protocol resists both private key leakage and replay of previously captured proofs. By introducing VRF-based randomness, the scheme simultaneously ensures uniqueness, unpredictability, and public verifiability of the effective nonce, thereby fundamentally blocking attacks that rely on nonce reuse, predictability, or random-number manipulation.

a:



b:

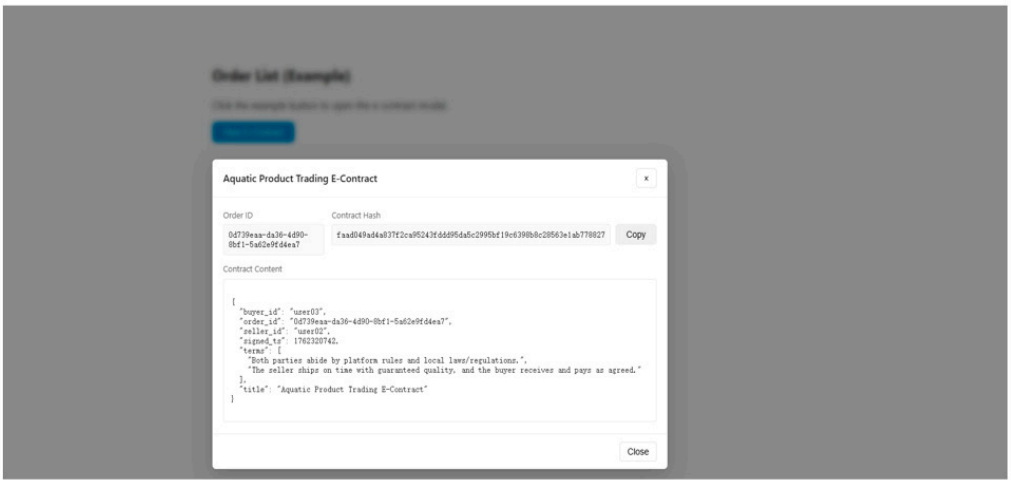


Figure 8. (**a**): Transaction rating; (**b**): transaction contract inquiry.

4.2.2. Anti-Abrasion Attack

Grinding attacks represent a core threat to VRF security, as adversaries may manipulate VRF inputs (seed) to obtain advantageous outputs. To counter such attacks, a commitment protocol is employed during the seed generation phase to guarantee bidirectional fairness and prevent unilateral manipulation by either the client or the server. Furthermore, once a seed is registered, it is committed and rendered immutable via blockchain recording, thereby significantly reducing the risk of leakage. Additionally, each seed attempt necessitates a complete registration process, rendering large-scale grinding attacks economically infeasible.

4.2.3. Protection Against Side-Channel Attacks

Side-channel attacks infer secret information by analyzing the physical characteristics of cryptographic operations, posing a significant security threat in practical deployments. This solution employs a multi-layered protection mechanism for critical cryptographic operations in the design and implementation of the VRF-Schnorr authentication protocol, effectively resisting timing and power analysis attacks. At the algorithmic level, the Montgomery ladder algorithm is employed to compute the commitment $R = r \cdot G$. Constant-time conditional swaps and a fixed sequence of point addition and doubling operations ensure that the execution time is completely independent of the bit values of the scalar r. This algorithm causes the power consumption trace of scalar multiplication to exhibit a fixed pattern, thereby eliminating pattern differences identifiable by Simple Power Analysis. At the underlying arithmetic level, when the response $s = r + c \cdot x (mod)\, q$ is calculated, all modular multiplication, modular addition, and modular subtraction operations are implemented in constant time, eliminating branching behavior based on operand size. At the data level, the introduction of a randomization factor b severs the direct correlation between the power consumption trace and the true private key x, significantly increasing the difficulty of Differential Power Analysis attacks. Even if high-quality power consumption traces can be collected by an attacker, an effective secret value correlation model cannot be established. At the implementation level, a security-audited cryptographic library (such as the secure mode of libsecp256k1) is used, and critical operations are deployed in a hardware security module to further enhance physical security.

4.2.4. Data Privacy Protection and Resistance to Keyword Guessing Attacks

Privacy protection is provided at the data level, and appropriate countermeasures are implemented to defend against keyword guessing attacks. For content encryption, the system uses AES-GCM authenticated encryption to protect sensitive regulatory data stored off-chain, ensuring that unauthorized parties cannot modify or read the plaintext. For order matching, users interact only with searchable encrypted ciphertext. They generate trapdoors (search tokens) and perform keyword matching directly over PEKS ciphertext, completing the retrieval without decrypting the underlying order content. Even if other nodes or adversaries obtain the stored ciphertext, they are unable to recover the plaintext order data. To reduce long-term key exposure risks, the system adopts a temporary session-key strategy for AES encryption: a fresh session key is generated for each transaction and expires after use. This limits the impact of potential key compromise and further enhances data privacy and forward security. To address the risk of keyword guessing—wherein adversaries may generate trapdoors by enumerating common keywords—salting and padding techniques are employed, with the salt value bound to the order ID to guarantee uniqueness. This approach prevents adversaries from precomputing the mapping from keywords to ciphertext, thereby enhancing the semantic security of PEKS schemes in open environments.

4.2.5. Anti-Syllabus Attack, Collusion Scoring Attack and Anti-Tampering Auditing

The system employs a reputation-based access control mechanism. The reputation module combines user behavior, transaction history, and social feedback to compute a comprehensive reputation value for each user. Orders are classified into five sensitivity levels (L1–L5) based on attributes such as price. Among the user behavior factors, a calculation threshold for ratings is established and a mutual rating detection mechanism is introduced to effectively suppress collusive scoring attacks. The user attribute dimensions in the reputation model are designed to fully account for heterogeneity in identity quality, while the behavioral dimensions, which are influenced by transaction amounts, help to

prevent Sybil attacks arising from the accumulation of reputation through numerous small transactions to a certain extent. After each transaction, the system updates the relevant users' reputation values and writes the new values into channel-3 as time-ordered reputation snapshots. All reputation calculations are executed on-chain by chaincode; clients cannot directly modify reputation data. Transaction-related ratings are protected by digital signatures from both parties, enabling the system to verify the source of each score and preventing forged or manipulated evaluations. Reputation snapshots are periodically aggregated by the SaveRepSnapshot chaincode to ensure consistency of reputation states across all nodes. Accordingly, at the reputation management level, security mechanisms such as anti-forgery, Sybil attack resistance, and anti-collusion scoring are implemented.

4.3. Performance Analysis

To comprehensively evaluate the actual performance of the proposed solution in an online aquatic product trading scenario, this section conducts a systematic evaluation from four dimensions: experimental environment configuration, identity authentication efficiency, on-chain resource consumption, and access control mechanism performance. Through controlled experiments and stress tests, the time overhead and economic cost of each module are quantitatively analyzed to verify the overall effectiveness of the solution.

4.3.1. Experimental Environment Configuration

To approximate a realistic deployment scenario, we adopt a distributed node deployment architecture. The hardware platform consists of high-performance commercial servers, and the software stack is built on mainstream blockchain development tools and frameworks, ensuring both reliability and reproducibility of the experimental results. The Hyperledger Fabric network is configured as a multi-channel, multi-organization consortium blockchain. The main channels and their roles are summarized in Table 5.

Table 5. Configuration Details.

Category	Configuration Item	Specification Details
Hardware Environment	Compute Nodes	Intel Xeon Gold 6348
	Storage Systems	2 TB NVMe SSD RAID 0
	Network Devices	Mellanox ConnectX-6 100GbE
Software Platform	Operating Systems	Ubuntu 22.04 LTS (Kernel 5.15)
	Container Environments	Docker 24.0.6, Kubernetes 1.28
Blockchain Environment	Underlying Blockchain	Ethereum Geth v1.13.0
	Smart Contracts	Solidity 0.8.18, Hardhat 2.17.0
	Development Tools	Web3.js 4.2.0, Truffle 5.8.0
Cryptographic Components	Core Algorithms	Python 3.9, Go 1.21
	Cryptography Libraries	OpenSSL 3.0.8, libsecp256k1 0.3.2

This configuration reflects the multi-channel access control model described in Section 3, and is used as the basis for the subsequent performance evaluation.

4.3.2. Time Consumption Analysis of Identity Authentication Scheme

As the first line of defense for system security, the performance of identity authentication directly impacts user experience. This scheme, based on the standard Schnorr protocol, introduces VRF (Virtual Randomization) to generate unpredictable random numbers, effectively enhancing the system's ability to resist replay attacks.

To quantify the performance overhead of the security enhancement mechanism, we conducted millisecond-level precision measurements on each stage of the authentication

process at a scale of 50 nodes. The test results are shown in Figure 9, illustrating the time consumption distribution of each stage under the minimum, average, and maximum states.

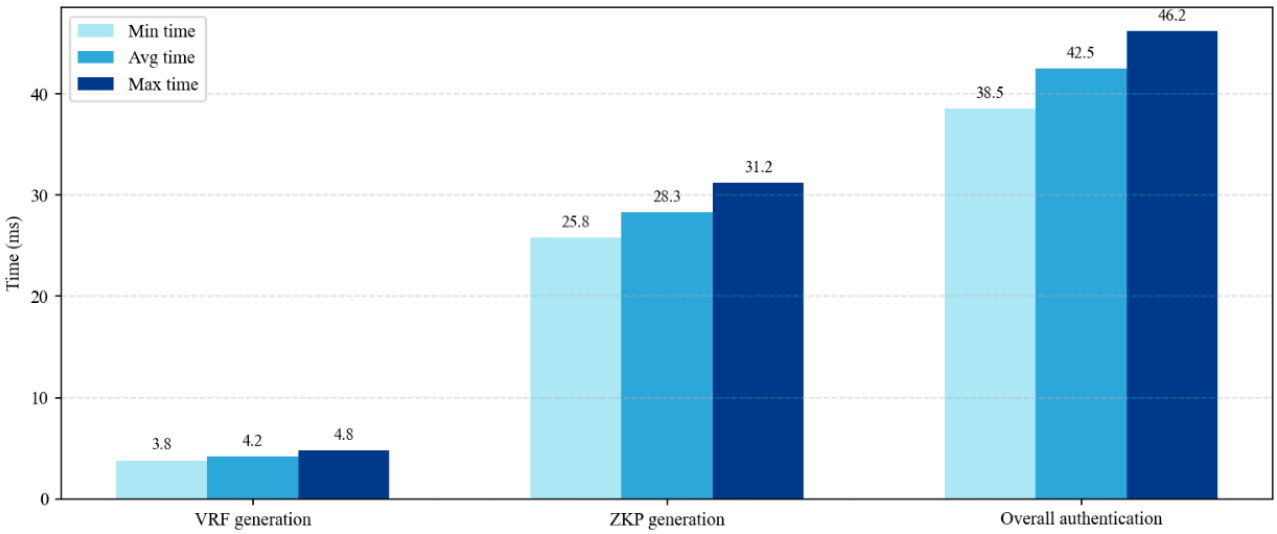


Figure 9. Time consumption distribution chart for each stage of identity verification.

Data analysis shows that the VRF random number generation stage has an average time consumption of 4.2 ms and a standard deviation of 0.32 ms, demonstrating good stability. Zero-knowledge proof generation, as the most computationally intensive stage, has an average time consumption of 28.3 ms, accounting for 66.6% of the total authentication time. It is noteworthy that the VRF security enhancement module contributes only about 21.9% of the time overhead, yet brings a significant security improvement, proving that this design achieves a good balance between security and performance.

4.3.3. Performance Analysis of Access Control Policy

While the introduction of access control mechanisms enhances the security of aquatic product transactions, it inevitably incurs additional system overhead. Therefore, a comprehensive evaluation from both performance and security perspectives is necessary. Existing research largely focuses on building decentralized ledgers and trusted traceability capabilities, but it still falls short in terms of the granularity, dynamism, and impact on matching efficiency of access control.

On the one hand, similar solutions to those examined in this study also construct matching models for aquatic product transactions based on blockchain and multi-chain collaboration mechanisms, thereby achieving decentralized storage of order data and tamper-proof ledger records. However, its matching process does not introduce explicit access control policies; the system assumes all participants have homogeneous permissions. While this design reduces matching latency, it struggles to effectively constrain low-reputation entities and cannot implement differentiated protection based on data sensitivity.

On the other hand, existing research has introduced certain forms of access control on the basis of decentralized architecture: some researchers have adopted an access control model based on identity and organizational roles, constraining read and write permissions of transaction records through pre-configured member identity information in the consortium blockchain, thus achieving permission isolation for different institutions; other researchers have introduced credit rating-driven transaction priority control on top of the consortium blockchain's identity and certificate mechanisms, prioritizing orders from high-reputation entities for matching, thus optimizing the reputation of participants and the matching order to some extent.

However, both of these solutions generally adopt static or semi-static access control strategies: permission configuration mainly relies on preset identity, organizational, or certificate information, lacking a dynamic perception and adjustment mechanism for transaction behavior characteristics and data sensitivity levels. Tightening or loosening permissions often requires manual intervention or a long governance cycle, making it difficult to support the real-time risk control needs of high-frequency trading scenarios.

To address the aforementioned issues, this solution, based on a decentralized ledger and multi-chain storage architecture, constructs an integrated dynamic access control mechanism encompassing reputation and data sensitivity. To analyze the actual impact of the above mechanism on system performance, this paper designed 1000 random order matching experiments to compare the performance of the basic system without access control mechanism and the proposed scheme on key indicators. The experimental results are shown in Table 6.

Table 6. Impact of access control mechanisms on system performance.

Performance Indicators	No Control Scheme	Functional Module	Performance Variation in This Scheme	Cause Analysis
Permission check time (ms)	0	6.8 ± 0.45	+6.8 ms	Access policy verification overhead
Reputation query time (ms)	0	1.2 ± 0.15	+1.2 ms	On-chain reputation value retrieval
Data encryption time (ms)	0	4.5 ± 0.32	+4.5 ms	PEKS encryption operation
Average matching time (ms)	18 ± 1.2	31.6 ± 1.8	+72.7%	Superimposed effect of security mechanisms
Matching success rate (%)	72 ± 3.5	98.5 ± 1.2	+36.2%	Fine-grained access control

The results show that, due to the introduction of permission checks, reputation queries, and data encryption, the average time per matching operation increases from approximately 18.0 ms to 31.6 ms. This corresponds to an increase of around 13.6 ms (about 72.7% relative increase). Permission checking accounts for a substantial portion of this additional cost, roughly half of the total increased time. Compared with schemes that rely only on identity/organizational attributes or simple credit priority, our strategy has more complex decision logic. However, by precomputing and caching reputation values, we avoid heavy computations on the matching critical path and keep the end-to-end latency within an acceptable range.

More importantly, fine-grained access control significantly improves the matching success rate. Under the same order volume and market parameter settings, the success rate rises from approximately 72.3% in the uncontrolled scheme to 98.5% with the proposed scheme, an increase of 26.2 percentage points. This indicates that, compared with schemes without access control, our design effectively filters out transaction pairs with insufficient reputation or mismatched permissions, reducing invalid matching attempts. Compared with strategies that rely solely on static identity/organization attributes or a single reputation priority, our scheme dynamically updates reputation and sensitivity labels and bases permission decisions on current behavior and risk status. As a result, it improves both the actual matching success rate and resource utilization, while ensuring compliance and security.

In summary, although the proposed access control mechanism increases the latency of each individual matching operation relative to the no-control baseline, it substantially increases the probability that each matching attempt succeeds and reduces the total system load and time overhead caused by retries. For application scenarios with high security and compliance requirements—such as high-value aquatic product trading—this

approach achieves a favorable trade-off, maximizing security benefits at an acceptable performance cost.

4.3.4. Multi-Channel Architecture

To evaluate the performance advantages of the proposed multi-channel architecture, we conducted comparative tests on a transaction contract information retrieval scenario. In a traditional single-chain model, all business data—including user registration, supply and demand orders, and transaction contracts—are stored on the same blockchain in time sequence. Executing a query requires traversing the chain block by block from the tail, and it is difficult to avoid interference from a large amount of non-target data. In contrast, the multi-channel architecture designed in this paper separates data of different business entities into independent channels. In contrast, the multi-channel architecture designed in this paper separates data belonging to different business entities and system functions into independent channels. This hierarchical storage strategy enables query requests to be directed exclusively to the corresponding functional channel, thereby significantly narrowing the overall search scope. Channel-1 serves as the registration and identity authentication channel, primarily storing user VRF outputs, ZKP data, vector commitment parameters, and other identity and key-related materials. Channel-2 is dedicated to order transactions and stores AES–PEKS–encrypted orders and transaction records. Channel-3 supports reputation evaluation and aggregation, centrally maintaining scoring data and reputation snapshots. Channel-4 is responsible for system auditing and global aggregation, recording logs, notifications, global secondary parameters, and other operational and monitoring data. An aggregation parameter Π_{global} is introduced, defined as the sum of the local aggregation parameters Π_i from each channel (Channel-2, Channel-3, Channel-4): $\Pi_{global} = \sum_{i=1}^{n} \Pi_i$, where n denotes the number of channels and Π_i is the local aggregation parameter of the i channel. Through periodic or event-driven global aggregation mechanisms, convergence and consistency of cross-channel states are achieved.

In the comparative experiments, contract query was selected as the test task. We deployed 100, 200, 400, 800, 1600, and 1800 transaction records on-chain and measured the average response latency for a single contract query at each scale. As shown in Figure 10, under the single-chain architecture, when the number of contracts increases from 100 to 1800, the query time grows from approximately 950 ms to about 5400 ms, exhibiting a clear linear growth trend. This is because, in the single-chain model, retrieving a target contract requires scanning all blocks that may contain user, order, and other non-contract data; as data accumulates, interference from irrelevant information causes the query overhead to grow continuously.

These results verify the efficiency benefits of the multi-channel architecture in data retrieval. By physically separating functional domains and leveraging parallel processing, the system not only reduces the latency of individual queries, but also maintains stable performance under large-scale data loads, providing reliable technical support for high-concurrency online aquatic product trading platforms.

This paper evaluates the throughput of the proposed aquatic product trading system under different levels of concurrent requests. In the test scenario, the number of concurrent transaction requests was varied from 100 to 800 tx/s, and the average system throughput during the stable phase was recorded, as shown in Figure 11. The horizontal axis represents the concurrency (tx/s), and the vertical axis represents the system throughput (tx/s). when the number of concurrent requests increases from 100 tx/s to 300 tx/s, the system throughput rises rapidly from approximately 120 tx/s to 230 tx/s, indicating that the system still has considerable processing headroom in the low-to-medium load range. When concurrency further increases to 400–500 tx/s, throughput remains at a high and stable level of about

232–234 tx/s, suggesting that the system is operating near its optimal processing capacity. As concurrency continues to 600–800 tx/s, throughput decreases slightly and stabilizes at around 223 tx/s. This degradation is mainly due to the additional computational and I/O overhead introduced by access control verification, reputation queries, and encryption operations under high load, causing a mild overload state. Nevertheless, the system still maintains high processing capacity and stability.

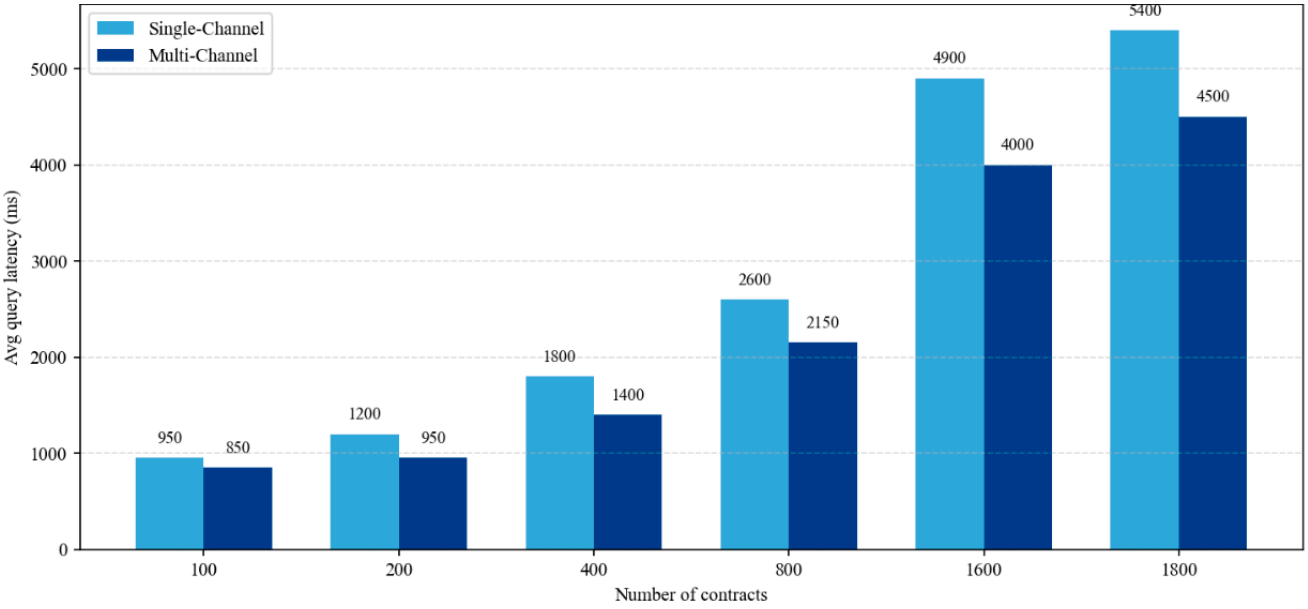


Figure 10. Contract query latency: single-channel vs. multi-channel.

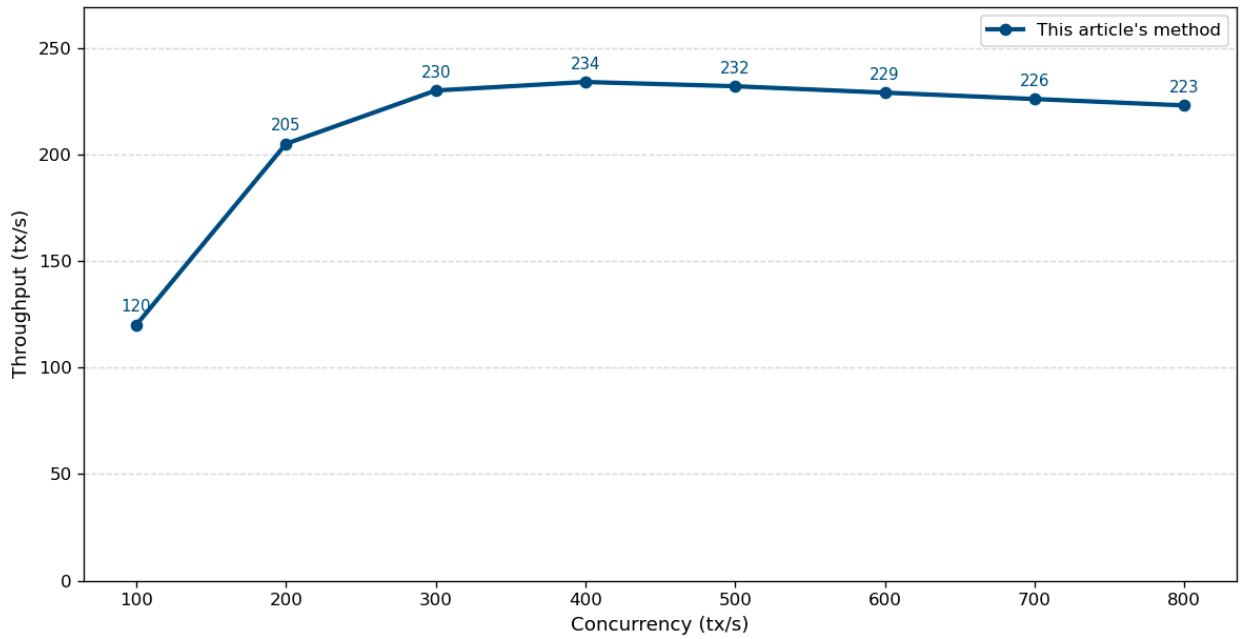


Figure 11. Throughput (TPS) analysis.

Overall, the test results show that, even after introducing reputation- and data-sensitivity-based access control mechanisms, the system can sustain a peak throughput of approximately 230 tx/s within a concurrency range of about 300–500 tx/s. This meets the high-concurrency and high-reliability requirements of practical aquatic product trading scenarios and confirms that the proposed model achieves a favorable balance between access control overhead and overall processing performance.

4.3.5. Validation in Simulated Real-World Scenarios

To bridge the gap between theoretical simulations and practical deployment, conditions reflecting various real-world scenarios were thoroughly considered, and the proposed blockchain system for aquatic product trade was evaluated accordingly. This validation focuses on multi-stakeholder collaboration, dynamic market traffic patterns, and system resilience under network impairments.

As shown in Figure 12, the system demonstrates strong scalability across varying organizational scales. In actual aquatic product trade, multiple entities such as wholesalers, logistics providers, and regulatory bodies must participate in the consensus process. Our results indicate that as the number of independent organizations scales from 2 to 32, the system maintains a robust throughput of over 170 tx/s, despite the increased communication overhead and consensus complexity. Furthermore, we simulated a 24 h dynamic traffic pattern mimicking real-world market hours with dual peaks at 10:00 a.m. and 6:00 p.m. The system processing rate effectively tracks the fluctuating user request volume and reaches a saturation point at approximately 230 tx/s. Notably, the slight fluctuations observed during the peak periods accurately reflect resource contention and hardware scheduling jitter under maximum load, confirming its ability to handle high-concurrency periods in busy trading windows.

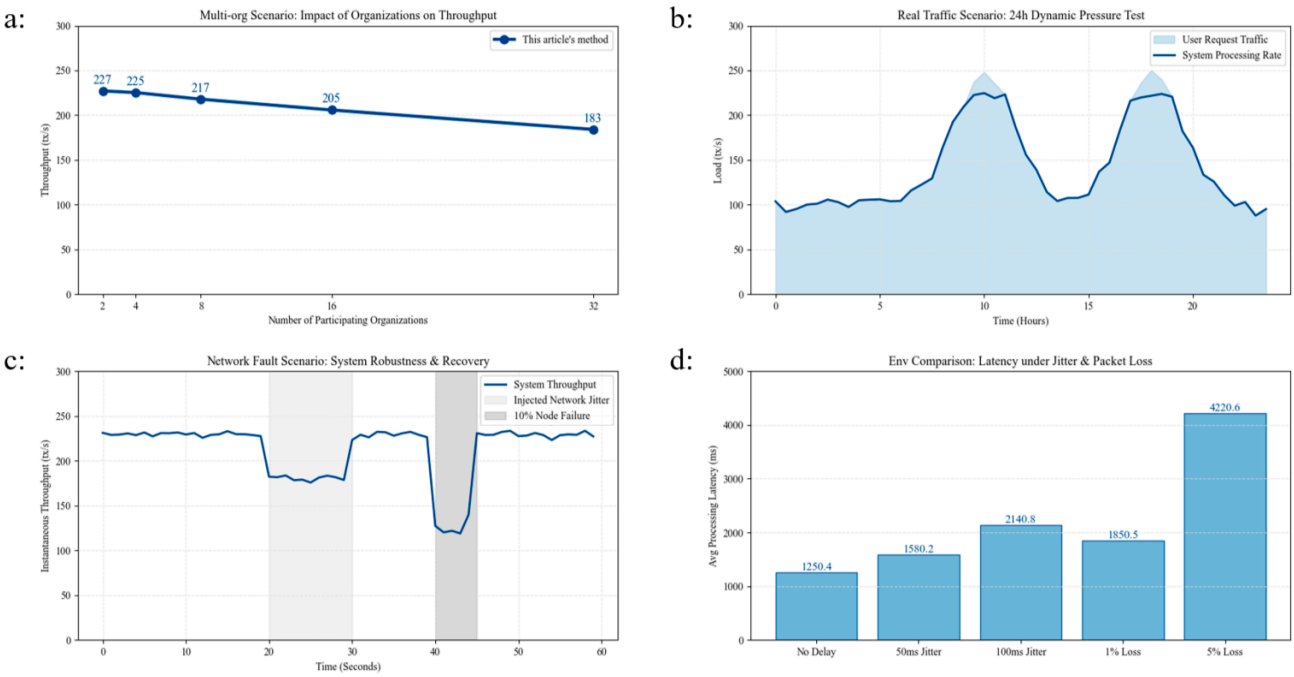


Figure 12. Performance and Robustness Verification of the System Under Diverse Highly Realistic Simulation Scenarios. (**a**): Impact of participant organization scale on system throughput; (**b**): evaluation of system processing rate under 24 h dynamic transaction workload; (**c**): throughput stability and recovery under network fluctuations and node failures; (**d**): comparison of average processing latency under diverse network constrained environments.

System robustness was further verified through fault injection and network degradation tests. During active operations, we injected network jitter and a 10% node failure to simulate unstable internet environments and hardware malfunctions. The system throughput experienced transient fluctuations but exhibited immediate self-healing capabilities, returning to normal processing levels as soon as the faults were mitigated. Additionally, we compared the average processing latency under different levels of network quality. Due to the high complexity of on-chain aquatic trade traceability and multifaceted contract

queries, the baseline processing latency is approximately 1250 ms. Under severe conditions such as 100 ms jitter or 5% packet loss, the latency increases to roughly 4220 ms. These results collectively demonstrate that the proposed framework remains operational and resilient under unpredictable network conditions, providing a stable foundation for the complex, multi-organizational environment of the aquatic product supply chain.

5. Conclusions

This paper addresses the core problems of trust deficiency, privacy leakage, and rigid access control in online aquatic product trading. Building on blockchain technology, several key security mechanisms are designed and integrated, leading to the following main contributions. First, a strong identity authentication protocol that combines a VRF with Schnorr ZKP is constructed. By using deterministic, verifiable random number derivation, this protocol significantly enhances resistance to replay and nonce-based attacks, while supporting decentralized and privacy-preserving identity verification. Second, a fine-grained access control mechanism based on a dynamic, multi-dimensional reputation model is designed. The mechanism quantifies user behavior, attributes, and historical transaction performance into a comprehensive trust value, and links this value to order sensitivity. In this way, it realizes dynamic, precise permission allocation, improves matching quality, and simultaneously protects data privacy. Third, an order privacy-preserving matching scheme that combines AES symmetric encryption with PEKS is proposed. This scheme ensures that transaction data remain end-to-end encrypted while still supporting efficient keyword retrieval and matching, thereby mitigating the risk of commercial information leakage during the matching process. Fourth, a prototype system with a multi-channel architecture is implemented on Hyperledger Fabric. By isolating business functions and leveraging parallel processing, the system increases throughput while maintaining audit transparency and protecting data privacy. Both theoretical analysis and experimental results have demonstrated that the proposed comprehensive security solution can systematically enhance the security, privacy, and usability of online aquatic product trading platforms.

While initial success has been achieved in the integration of key technologies and system implementation, several directions remain to be explored. At the technical level, cross-chain identity verification and data exchange protocols need to be introduced to support interoperability across ecosystems. Future work should investigate lightweight cross-chain oracles and standardized data schemas to enable seamless data exchange while ensuring security and decentralization. Furthermore, searchable encryption technologies that support range queries and fuzzy queries should be explored, detailed incentive and governance mechanisms for different stakeholders should be designed, and adaptive learning algorithms should be introduced to enhance the accuracy and robustness of reputation models. At the performance level, more efficient ciphertext matching algorithms and on-chain storage compression strategies need to be developed to address the scalability challenges imposed by large-scale transaction data. At the application and regulatory levels, compliance remains critical. Further analysis and enhancement of the compatibility between blockchain-based transaction systems and existing legal frameworks are required, and active participation in the development of industry technical standards is essential to promote standardized deployment and large-scale adoption of the technology.

Author Contributions: Conceptualization, W.W., L.Y. and M.C.; methodology, L.Y. and Y.G.; software, Y.Z., and M.C.; validation, W.W. and M.C.; formal analysis, L.Y. and Y.Z.; investigation, M.C. and Y.G.; writing—original draft preparation, L.Y. and W.W.; writing—review and editing, L.Y., W.W. and M.C.; visualization, W.W. and M.C.; funding acquisition, M.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Research and Development Planning in Key Areas of Guangdong Province, NO. 2021B0202070001.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wang, J.; Wang, A.; Zhu, L.; Wang, H. The Effect of Product Image Dynamism on Purchase Intention for Online Aquatic Product Shopping: An Eeg Study. *Psychol. Res. Behav. Manag.* **2021**, *14*, 759–768. [CrossRef]
2. Tiwari, K.; Kumar, S. Accelerating Healthcare Data Access: A Blockchain and Decentralized Indexing Approach. *Concurr. Comput.* **2025**, *37*, e70172. [CrossRef]
3. Khan, A.A.; Alsufyani, A.; Alsufyani, N.; Mohamed, M.A. BAML: A Decentralized Approach to Secure, Privacy-Preserving Financial Compliance for Enhancing Anti-Money Laundering with Blockchain Hyperledger and Federated Learning. *Peer Peer Netw. Appl.* **2025**, *18*, 270. [CrossRef]
4. Werapun, W.; Karode, T.; Suaboot, J.; Arpornthip, T.; Sangiamkul, E. NativeVRF: A Simplified Decentralized Random Number Generator on EVM Blockchains. *Systems* **2023**, *11*, 326. [CrossRef]
5. Zhang, L.; Zhou, F.; Zhang, Q.; Xiong, W.; Tian, Y. An Improved Secure Designated Server Certificateless Authenticated Searchable Encryption Scheme for IIoT. *High-Confid. Comput.* **2025**, *5*, 100301. [CrossRef]
6. Gao, X.; Zhang, Y. Advancing Remote Consultation through the Integration of Blockchain and Ant Colony Algorithm. *Math. Biosci. Eng.* **2023**, *20*, 16886–16912. [CrossRef]
7. Chen, Y.; Ou, W.; Pang, M.; Ma, J.; Yue, Q.; Han, W. A Cross-Chain Model for Warehouse Receipts in Port Supply Chain Based on Notary Mechanism and ShangMi Cryptographic Algorithms. *Sci. Rep.* **2025**, *15*, 14390. [CrossRef]
8. Du, Z.; Gong, J.; Qian, H. LBFT: An Asynchronous Committee-Based Blockchain Storage Strategy on Zero Trust Model. *Appl. Sci.* **2022**, *12*, 11790. [CrossRef]
9. Liujie, J.; Yan, S.; Hao, Q. Research on Vaccine Cold Chain Monitoring System Based on RFID Flexible Temperature Tag and 5G. *J. Big Data Comput.* **2024**, *2*, 29–32. [CrossRef]
10. Zhang, Y.; Liu, Y.; Jiong, Z.; Zhang, X.; Li, B.; Chen, E. Development and Assessment of Blockchain-IoT-Based Traceability System for Frozen Aquatic Product. *J. Food Process Eng.* **2021**, *44*, e13669. [CrossRef]
11. Yuan, X.; Pei, H.; Wu, Y. Three-Level Intrusion Detection Model for Wireless Sensor Networks Based on Dynamic Trust Evaluation. *Comput. Mater. Contin.* **2025**, *84*, 5555–5575. [CrossRef]
12. Qiao, Z.; Yu, Y.; Zhou, Y.; Zheng, D. A Continuous Leakage-Amplified IBE Scheme with Perfect Key Update. *Theor. Comput. Sci.* **2025**, *1055*, 115545. [CrossRef]
13. Yang, R.; Zhang, Q.; Meng, L.; Wang, C.; Hu, Y. LUTBIO: A Comprehensive Multimodal Biometric Database Targeting Middle-Aged and Elderly Populations for Enhanced Identity Authentication. *Inf. Fusion.* **2025**, *118*, 102945. [CrossRef]
14. Fu, Q.; Ang, Z.; He, F.; Xu, P.; Yang, G. A Secure Operation Method Based on Secure Chip Identity Authentication. *Autom. Control Comput. Sci.* **2025**, *59*, 417–425. [CrossRef]
15. Saeidi, M.R.; Mala, H. Post-Quantum Anonymous Password Authenticated Key Exchange with Indirect Public Key Validation. *J. Supercomput.* **2025**, *81*, 1325. [CrossRef]
16. Zhang, L.; Sun, Y.; Xu, Y.; Zhu, H. IDMS Quantum Password-Authenticated Key Exchange Protocols. *Int. J. Inf. Comput. Secur.* **2024**, *25*, 351–366. [CrossRef]
17. Sakwa, C.O.; Anyembe, A.O.; Li, F. A Survey of Folding-Based Zero-Knowledge Proofs. *Inf. Sci.* **2026**, *724*, 122698. [CrossRef]
18. Wang, W.; Zhang, X.; Chen, M.; Zou, Y.; Ge, Y. Trading Matching Model and System Implementation for Aquatic Products Based on Blockchain. *Trans. Chin. Soc. Agric. Mach.* **2023**, *54*, 364–375.
19. Zou, Y.; Jing, H.; Chen, M.; Ge, Y.; Wang, W. Cross-Chain Trading Model of Aquatic Products Based on Notary Mechanism. *Trans. Chin. Soc. Agric. Mach.* **2024**, *55*, 262–271.
20. Wang, W.; Wang, H.; Chen, M.; Zou, Y.; Ge, Y. Aquatic Product Trading Matching Model Based on Multi-Chain Storage Optimization. *Trans. Chin. Soc. Agric. Mach.* **2024**, *55*, 272–283.
21. Yan, Y. Optimization of Import and Export of Aquatic Products in Zhanjiang City—Platform Construction Based on Blockchain Perspective. *Appl. Math. Nonlinear Sci.* **2024**, *9*. [CrossRef]
22. Ge, Y.; Zheng, K.; Zou, Y.; Chen, M.; Wang, W. Enhanced HACCP Credibility and Visualization Traceability Model. *Trans. Chin. Soc. Agric. Mach.* **2023**, *54*, 385–396, 411.
23. Liu, C.; Xu, J.; Zhao, Z.; Chen, S.; Li, Y.; Zhang, X. Food Full-Process and All-Information Traceability Based on Multi-Chain Blockchain and Trusted Transmission Protocols. *Expert. Syst. Appl.* **2025**, *286*, 128008. [CrossRef]

24. Rossi, S.; Gemma, S.; Borghini, F.; Perini, M.; Butini, S.; Carullo, G.; Campiani, G. Agri-Food Traceability Today: Advancing Innovation towards Efficiency, Sustainability, Ethical Sourcing, and Safety in Food Supply Chains. *Trends Food Sci. Technol.* **2025**, *163*, 105154. [CrossRef]
25. Sun, C.; Sun, Z.; Luo, N.; Xing, B.; Wang, S.; Gao, G.; Yang, X. Cross Domain Traceability and Regulatory Data Sharing Method for Food Supply Chain Based on Blockchain. *Trans. Chin. Soc. Agric. Mach.* **2025**, *56*, 33–46.
26. Liu, X.; Luo, N.; Xing, B.; Gao, G.; Sun, M.; Sun, C. Attribute-Based Searchable Encrypted Agricultural Blockchain Traceability Private Data Access Control Method. *Trans. Chin. Soc. Agric. Mach.* **2024**, *55*, 433–443.
27. Sun, F.; Wang, P.; Zhang, Y.; Kar, P. BFSCM: An Enhanced Food Supply Chain Management System Using Hybrid Blockchain and Recommender Systems. *Blockchain: Res. Appl.* **2025**, *6*, 100245. [CrossRef]
28. Wang, W.; Teng, D.; Chen, M.; Ge, Y.; Zou, Y. A Trading Matching Model for Aquatic Products Based on Blockchain and Credit Mechanisms. *Math. Biosci. Eng.* **2023**, *20*, 19732–19762. [CrossRef] [PubMed]
29. Yan, L.; Ge, L.; Wang, Z.; Zhang, G.; Xu, J.; Hu, Z. Access Control Scheme Based on Blockchain and Attribute-Based Searchable Encryption in Cloud Environment. *J. Cloud Comput.* **2023**, *12*, 61. [CrossRef]
30. Sharma, K.; Kumar, N.; Kaushal, R.K.; Verma, A. Privacy and Trustworthiness in Healthcare: A Blockchain and Attribute Based Access Control Integrated Approach. *SN Comput. Sci.* **2025**, *6*, 857. [CrossRef]
31. Singh, J.; Sinha, A.; Goli, P.; Subramanian, V.; Shukla, S.K.; Vyas, O.P. Insider Attack Mitigation in a Smart Metering Infrastructure Using Reputation Score and Blockchain Technology. *Int. J. Inf. Secur.* **2022**, *21*, 527–546. [CrossRef]
32. Ke, Z.; Cao, Y.; Chen, Z.; Yin, Y.; He, S.; Cheng, Y. Early Warning of Cryptocurrency Reversal Risks via Multi-Source Data. *Financ. Res. Lett.* **2025**, *85*, 107890. [CrossRef]
33. Liu, L.; Gu, Q.; Ke, W. AGTS: Novel Automated Generation of Smart Contract Test Suites for Hyperledger Fabric. *Sci. Rep.* **2025**, *15*, 29675. [CrossRef]
34. Cai, C.; Peng, C.; Tian, Y.; Tan, W.; Niu, J.; Teng, F. BCDAP-DGS: Dynamic Group Signature and Batch Cross-Domain Authentication Protocol for Intelligent Transportation. *IEEE Trans. Intell. Transp. Syst.* **2025**, *26*, 18680–18694. [CrossRef]
35. Chen, Z.; Zhang, H.; Mu, Q.; Li, D.; Zhu, H. In-Vehicle Device Data Tampering Detection: Accurate Identification Based on Correlation Calculation and Data Relationship. *Comput. Secur.* **2025**, *158*, 104648. [CrossRef]
36. Saif, A.; Abbas, S.; Fayed, Z. The PDBO Algorithm for Discrete Time, Cost and Quality Trade -off in Software Projects with Expressing Quality by Defects. *Procedia Comput. Sci.* **2015**, *65*, 930–939. [CrossRef]
37. Cai, C.; Peng, C.; Tang, H.; Xiao, B.; Tan, W. DWTAT-DASIS: Fusion of Discrete Wavelet Transform and Access Tree for Distributed Authentication in Secret Image Sharing. *Comput. Stand. Interfaces* **2025**, *93*, 103969. [CrossRef]
38. Wan, L.; Cao, Y.; Bi, C.; Liu, Z.; Ma, J.; Ren, Y. PBRU: Privacy-Preserving and Blockchain-Assisted Reputation Updating with Malicious Detection for Cloud-Supported Vehicular Networks. *IEEE Trans. Inf. Forensics Secur.* **2025**, *20*, 13026–13040. [CrossRef]
39. Sovyn, Y.; Khoma, V.; Podpora, M. Comparison of Three CPU-Core Families for IoT Applications in Terms of Security and Performance of AES-GCM. *IEEE Internet Things J.* **2020**, *7*, 339–348. [CrossRef]
40. Bi, C.; Cao, Y.; Ren, Y.; Tian, Y.; Wan, L.; Ke, W. A Blockchain-Enabled Secure Authentication and Fair Trading Scheme for Shared Charging Systems. *IEEE Trans. Netw. Sci. Eng.* **2025**, *12*, 110992–111006. [CrossRef]
41. Liao, T. Blockchain-Enabled Police Management Framework for Securing Police Data. *Soft Comput.* **2023**, *27*, 18061–18075. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.