

Article

A Dynamic Trust Evaluation and Risk Control Mechanism for Heterogeneous Cross-Chain Nodes

Zepeng Chen ¹, Hui Liu ^{1,*}, Lin Zhang ¹ and Chenjie Wu ²

¹ School of Information and Control Engineering, China University of Mining and Technology, Xuzhou 221116, China; tb22060001a41@cumt.edu.cn (Z.C.); lin.zhang@cumt.edu.cn (L.Z.)

² School of Economics and Management, China University of Mining and Technology, Xuzhou 221116, China; ts23070344p31@cumt.edu.cn

* Correspondence: iipl_peng@cumt.edu.cn

Abstract

Existing cross-chain bridges over-rely on static collateralization and post-event penalties, leaving them vulnerable to concealed on-off attacks and rational group collusion. To address these limitations, this paper proposes a Dynamic Trust Evaluation and Risk Control (DTERC) mechanism for heterogeneous cross-chain relay nodes. First, DTERC develops a multidimensional trust quantification model that combines temporal decay, robust multi-observer latency aggregation, verification accuracy, online stability, and an asymmetric one-strike penalty triggered only by cryptographic evidence. Second, DTERC constructs a threshold-aware N-player evolutionary game model to characterize the k-of-N signature structure of cross-chain relay consensus and introduces a dynamic staking function to reduce the economic incentive for collusion under bounded attack-value and parameter conditions. Third, DTERC designs a threshold-preserving FastPath mechanism to reduce redundant verification for low-risk transactions while retaining committee-level confirmation and challenge-based fallback. The empirical evaluation combines multi-agent simulation, smart-contract prototype testing, whitelist-compromise stress tests, malicious-oracle robustness analysis, network-jitter experiments, repeated trials, and parameter-sensitivity analysis. The results show that, under the tested settings, DTERC reduces the malicious transaction success rate to 0.15% under a 50% initial collusion scenario, lowers core contract Gas overhead by 35.7%, and reduces average end-to-end latency by approximately 10% in benign FastPath conditions. These findings indicate that DTERC improves the security–efficiency trade-off of heterogeneous cross-chain relay networks while making its assumptions and limitations explicit.

Keywords: cross-chain interoperability; dynamic trust evaluation; evolutionary game theory; on-off attack; dynamic staking; risk control



Academic Editor: Hossain Shahriar

Received: 12 May 2026

Revised: 11 June 2026

Accepted: 15 June 2026

Published: 17 June 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 \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

With the vigorous development of blockchain technology, the industry has fully transitioned into the “Multi-chain Era”. Heterogeneous blockchains, such as Ethereum and Hyperledger Fabric, play irreplaceable roles within their respective financial, governmental, and Internet of Things (IoT) ecosystems. However, inherent differences in underlying consensus mechanisms and cryptographic primitives have resulted in severe “value islands.” Cross-chain interoperability has therefore become core infrastructure for breaking these boundaries and enabling the seamless transfer of assets and data [1]. Depending on the

detailed architectural design, mainstream cross-chain technologies mainly include notary schemes, hash-locking, and relay chains, which have been widely adopted in Decentralized Finance (DeFi) applications [2].

Although these bridges introduce immense liquidity, they also create highly vulnerable attack surfaces. Many exploits have targeted these bridges in recent years, causing losses of billions of dollars. According to Systematization of Knowledge (SoK) reviews, most cross-chain hazards do not stem from the cryptographic primitives of the individual chains but rather from failed “trust assumptions” in cross-chain relay networks or external interaction modules [3–5]. A typical heterogeneous cross-chain architecture assumes that relay nodes monitor source-chain events and submit cryptographic proofs to the destination chain. Therefore, the honesty of their behavior directly affects the security benchmark for cross-chain assets.

Popular decentralized cross-chain solutions tend to use static over-collateralization (Proof of Stake, PoS) rules to constrain relay nodes. This paradigm relies on the explicit assumption of “static complete rationality.” Yet, in complex real networks, nodes not only face computational and informational restrictions but are also tempted by large cross-chain liquidity pools and evolve into “rational adversaries.” Static assessment and after-the-fact slashing have two fatal flaws. First, a unidimensional assessment cannot detail a node’s past interaction quality. Second, they are highly susceptible to “on–off attacks” (or traitor attacks). In such scenarios, malicious nodes initially behave honestly to accumulate high credit and pool funds, then suddenly collude to forge valid proofs during a high-value cross-chain transaction. Because the illicit profit far exceeds the fixed static staking penalty, the economic deterrence of traditional PoS is entirely bypassed [6].

Researchers have recently deployed continuous trust evaluation mechanisms in areas such as the Internet of Vehicles (IoV) and Federated Learning (FL) to tackle node trust management in distributed networks. By calculating the dynamic trustworthiness of entities based on multidimensional behavior weights, these methods have demonstrated strong anti-collusion and anti-defamation performance [7,8]. Nevertheless, directly introducing these heuristic, pure-reputation-based mechanisms into high-frequency, high-value heterogeneous cross-chain environments is not feasible. Cross-chain systems require strict financial finality; therefore, pure reputation models do not have the rigorous mathematical foundation to counter high-complexity, rational adversarial behaviors, as well as the risk circuit-breaking loop that must be tightly coupled with the underlying economic consensus.

Evolutionary Game Theory (EGT) provides a suitable mathematical framework to address dynamic cross-chain node trust evolution and malicious behavior prevention. EGT relaxes the strict “complete rationality” assumption of classical game theory and assumes that network nodes dynamically adjust their strategies through “trial, imitation, and learning.” This premise aligns with the profit-driven interaction patterns of validators in the Web3.0 ecosystem [9,10].

Therefore, this study proposes a Dynamic Trust Evaluation and Risk Control (DTERC) mechanism to address the fragility of static staking mechanisms and node behavior uncertainty in heterogeneous cross-chain networks. The paper’s main contributions can be summarized as follows:

1. Proposing a dynamic trust quantification model incorporating temporal decay and asymmetric penalties. Moving beyond the constraints of traditional static staking bridges, we build a multidimensional continuous trust monitoring system with features such as robust multi-observer response latency, verification accuracy, and online stability. The model estimates the latency component through median aggregation over multiple independent observers, preventing a single biased observation point from unfairly penalizing honest nodes. In particular, the model addresses on–off attacks by introducing a

reputation temporal decay factor and a one-strike veto method supported only by cryptographic evidence. This asymmetric penalty mechanism reduces the advantage of exploiting historical reputation and immediately isolates nodes once the system confirms verifiable malicious evidence.

2. Formulating a threshold-aware EGT-based dynamic staking and hierarchical risk control loop. We construct an N-player threshold evolutionary game model for the k-of-N signature structure of cross-chain consensus and incorporate the threshold breach probability into the evolutionary payoff function. The proposed mechanism uses a risk premium lever to reconstruct the payoff matrix and reduce the expected profitability of collusion under specified attack-value and parameter boundaries. It further designs a finite-state-machine-based hierarchical risk control channel, allowing FastPath to improve efficiency only under threshold-preserving and challengeable verification conditions.

3. Conducting extended empirical evaluations to assess security and prototype-level feasibility. Based on multi-agent simulations and an Ethereum/Fabric prototype, we perform adversarial security testing, FastPath compromise stress testing, multi-oracle robustness analysis, network-jitter evaluation, repeated trials with confidence intervals, and parameter sensitivity analysis. The experimental results indicate that, compared with Static-PoS, Beta-Reputation, and DDRV baselines, DTERC reduces the malicious transaction success rate (MTSR) to 0.15% under a 50% initial collusion setting while reducing Gas overhead and latency under benign FastPath conditions.

2. Related Work

The rapid evolution of the multi-chain ecosystem has propelled cross-chain interoperability to the forefront of both academic and industrial research. However, the frequent and devastating hacks targeting cross-chain bridges in recent years have exposed critical security vulnerabilities. Comprehensive surveys by Duan et al. [11] and Belchior et al. [12] emphasize that most cross-chain asset losses stem not from breaking the cryptographic algorithms of individual chains, but rather from the failure of “trust assumptions” within the relay layer and external interaction vulnerabilities of smart contracts. Targeting these unique vulnerabilities, Lee et al. [13] and Notland et al. [14] conducted in-depth SoK studies on massive bridge exploits. Their analyses reveal the highly concealed “on-off attacks” and rational collusion behaviors plaguing cross-chain networks. Based on this threat landscape, Zhou et al. [15] put forward BridgeGuard, a framework that detects vulnerabilities to external interactions in bridge router contracts using symbolic dataflow analysis. At the same time, to improve the traceability of decentralized bridge applications, Lin et al. [16] have proposed CONNECTOR, a solution that provides automated cross-chain transaction association. These studies expand the security boundaries of cross-chain bridges to a large extent, but their focus is mainly limited to post-event auditing, code vulnerability mining, and transaction tracing. On the other hand, mainstream industrial solutions continue to apply a fixed static collateralization (Static-PoS) and include passive, post-incident penalization. As of now, no proactive ex ante control mechanisms have been proposed to target behavioral trustworthiness in the cross-chain relay nodes directly.

Continuous trust evaluation mechanisms have gradually been integrated into blockchain architectures to restrict malicious behavior in distributed networks. Hu et al. [17] were among the first to introduce it into the cross-chain environment: they proposed a decentralized dynamic reputation value (DDRV) assessment method that reduces cross-chain operational risks by quantifying the historical behaviors of notaries. Trust mechanisms have also advanced considerably in other domains of decentralized security: Gu et al. [18] designed an efficient decentralized trust management system for intelligent transportation based on consortium blockchains; Masango et al. [19] proposed a redactable blockchain

mechanism that supports dynamic trust for malicious tampering prevention; and Shao et al. [20] presented a multidimensional data trust evaluation framework for blockchain-based IoT systems that relies on machine learning approaches. While such studies are useful references for node admission to decentralized environments, cross-chain relay networks are distinguished by very high asset sensitivity and extreme profit-driven incentives. Thus, classical pure mathematical reputation accumulation models (such as the Bayesian Beta-Reputation) or linear penalty mechanisms based on time windows (e.g., DDRV [17]) show obvious shortcomings. Most importantly, they show serious “defensive lag” when confronted with premeditated on–off attacks. Because they lack deep coupling with underlying staking capital and smart contract circuit-breaking logic, they cannot provide a meaningful amount of economic deterrence to rational adversaries.

Motivated by the rational behavior of participants and profit-seeking within the blockchain ecosystem, an essential theoretical approach for a formal investigation of endogenous security has evolved in the form of EGT. EGT relaxes the “complete rationality” strict assumption. Liu et al. [21] established an EGT model of the “Validator’s Dilemma” in PoW mining pools and discovered that pool strategy selection is significantly influenced by penalty mechanisms. Sun et al. [22] used evolutionary games to create new incentive mechanisms for redactable blockchain governance and mathematically proved that the implementation of well-designed rewards and punishments can drive system security convergence. Furthermore, Zhang et al. [23] formulated an EGT model for data supply chains based on prospect theory. Li et al. [24] employed evolutionary games in supply chain finance and proved that the mechanisms of blockchain guarantee results in the substantial suppression of default behaviors. Thus, these particular investigations prove EGT’s mathematical capability in multidimensional modeling of complex strategic interactions of blockchain nodes. Nevertheless, none of the existing research has deeply integrated EGT into the multidimensional dynamic trust evaluation of heterogeneous cross-chain networks to actively dismantle group collusion through the reconstruction of the adversarial payoff matrix.

Research on cross-chain transaction security and trust mechanisms has made progress, but it is still limited in the following respects: (1) Existing cross-chain bridge risk controls mainly use static over-collateralization and lack proactive defense based on multidimensional node behavior and real-time circuit-breaking capability during ex ante and in-process stages. (2) Current distributed trust calculation models are mostly heuristic, pure-reputation designs. When countering highly destructive “on–off attacks” and “rational collusion,” they lack asymmetric penalty constraints and rigorous mathematical proofs of evolutionary convergence. To bridge this research gap, this paper seeks to fundamentally elevate the endogenous security of cross-chain networks. By fusing time-decay evaluation, asymmetric penalties, and EGT, we construct a closed-loop dynamic trust and risk control mechanism tailored for heterogeneous cross-chain nodes.

3. System Modeling and Problem Formulation

Before delving into the proposed DTERC mechanism, this section first mathematically defines the heterogeneous cross-chain infrastructure and the specific threat models underpinning our research. Subsequently, utilizing EGT, we formalize the design objectives and the core problem of the security mechanism. For clarity and consistency in subsequent derivations, Table 1 summarizes the primary mathematical notations used in this paper.

Table 1. Notations.

Symbol	Description
$\mathcal{N}, N$	The set of cross-chain relay nodes $\mathcal{N} = \{n_1, n_2, \dots, n_N\}$, and the total number of nodes N .
C_{src}, C_{dst}	The source chain and destination chain in the heterogeneous cross-chain network.
k	The statutory consensus threshold for cross-chain message aggregate signatures, satisfying $k \leq N$.
$x(t)$	The proportion of nodes adopting the honest strategy at time t , where $x(t) \in [0, 1]$.
s_H, s_M	The pure strategy space of nodes: honest strategy (s_H) and malicious strategy (s_M).
U_H, U_M	The expected payoffs for a node adopting the s_H and s_M strategies, respectively.
$T_i(t)$	The dynamic trust value of the node n_i at time t .
$D_{req}(t)$	The dynamic staking requirement for a node to participate in cross-chain consensus.

3.1. Heterogeneous Cross-Chain Relay System Model

Contemporary heterogeneous cross-chain interoperability protocols, for example, LayerZero and Wormhole, generally implement relay-network-based architectures paired with lightweight smart contracts [25]. By deriving their common features, we develop the underlying cross-chain system architecture. Consider a cross-chain network that connects a source chain C_{src} with a destination chain C_{dst} , both of which have consensus mechanisms and state machines that vary. The relay nodes' decentralized set serves as the foundational layer that enables cross-chain communication:

$$\mathcal{N} = \{n_1, n_2, \dots, n_N\} \quad (1)$$

The process of the established architecture can be logically modeled into three steps:

(1) Event Listening and Capturing: The user u initiates a cross-chain transaction tx_{src} on C_{src} , invoking the source-chain gateway contract to lock or burn assets. A relay node $n_i \in \mathcal{N}$ captures the cross-chain message m by continuously monitoring the block headers and event logs of C_{src} .

(2) Multi-Party Consensus and Signing: The relay node verifies the legitimacy of message m (i.e., whether tx_{src} has achieved finality). Upon successful verification, node n_i signs the message digest using its private key sk_i , generating an independent signature share $\sigma_i = \text{Sign}(sk_i, \text{Hash}(m))$.

(3) Aggregation and Execution: Once the number of collected valid signatures reaches the predefined statutory threshold k ($k \leq N$), an aggregate or multi-signature Σ is generated. The relayer submits the proof payload $P = \langle m, \Sigma \rangle$ to the smart contract on C_{dst} . The destination contract verifies Σ against the public key set of the relay network; if valid, it executes the asset minting or releasing operations.

In the aforementioned traditional model, a node n_i typically acquires verification privileges only after depositing a fixed amount of stake [26], D_{stake} . Should a verifiable malicious act occur, this stake is slashed.

3.2. Threat Model and Attack Scenarios

It is essential to define the threat model of the proposed mechanism in detail, splitting it into cryptographic and economic/behavioral threats. This section sets the defense boundaries explicitly.

3.2.1. Cryptographic Assumptions

This paper assumes that the underlying consensus mechanisms of the heterogeneous blockchains ($\mathcal{C}_{src}, \mathcal{C}_{dst}$) are secure, and the applied cryptographic primitives, such as SHA-256 hash functions, ECDSA/BLS digital signatures, are computationally unforgeable. As a result, an attacker cannot reconstruct a node's private key or forge single-chain ledger records in polynomial time. Under no circumstances will the proposed mechanism tackle the situations when the fundamental cryptographic primitives are broken.

3.2.2. Economic and Behavioral Threats

To counteract high-level behavioral threats, the defense boundary is defined within the cross-chain relay layer failure of trust assumptions. The attackers are assumed to have the behavior of a Rational Adversary, that nodes' behaviors are driven by the desire to maximize their expected illicit profits and are not bound by fixed frequencies or patterns. In particular, three typical advanced persistent threats must be well protected against:

Rational Collusion Attack: Attackers cooperate by bribing, controlling, or uniting many nodes in the relay network. Suppose an attacker controls c nodes. When $c \geq k$ (the statutory threshold), the attacker can submit a forged cross-chain proof (e.g., minting assets out of thin air) to the destination chain. Let the expected illicit gain from this attack be V_{attack} . In traditional static models, if $V_{attack} > c \cdot D_{stake}$, a group of rational nodes will switch to collusion and attack the honest one in the traditional static model.

On-off/Traitor Attack: This represents the most destructive and elusive threat in cross-chain bridges [27]. A malicious node disguises itself as an honest strategy s_H over an initial time interval $t \in [0, T_{switch})$, exhibiting extremely high online rates and low latency. This behavior deceives the system into granting high credit evaluations and allocating substantial cross-chain order flows. However, at the precise moment $t = T_{switch}$ when processing a massive cross-chain transaction (e.g., $tx_{value} \gg D_{stake}$), the node abruptly switches to strategy s_M , executes the attack, reaps windfall profits, and subsequently exits the network.

Selective Relaying and Bad-mouthing: Driven by related interests, a dishonest node attempts to delay or drop cross-chain requests for selected users during the aggregate signature phase. Alternatively, it refuses to sign for honest nodes on purpose to degrade reputation within the system and to compete unfairly.

Sustained Multi-stage APT Attack: In addition to isolated on-off and collusion attacks, a rational adversary may conduct a long-horizon attack campaign across multiple stages. Such an APT may first accumulate reputation through compliant behavior, then probe latency and routing weaknesses, bribe or coordinate a subset of validators, fragment attack liquidity across several transactions, and finally launch a threshold-breach attempt when the expected return is highest. The adversary may dynamically allocate its financial strategy among bribery, opportunity cost manipulation, temporary over-collateralization, liquidity splitting, and delayed exit. Therefore, the threat model treats malicious behavior as a prolonged financial optimization process rather than a single-shot deviation.

DTERC distinguishes malicious cryptographic evidence from soft network-quality evidence. High latency, jitter, packet loss, or temporary reachability failure alone is not treated as cryptographically confirmed malicious behavior. These signals may reduce routing priority or trigger SlowPath fallback, but they cannot trigger one-strike veto unless accompanied by invalid signatures, forged proofs, or conflicting cross-chain messages.

3.3. Evolutionary Game Model and Problem Formulation

3.3.1. Evolutionary Game Model

Traditional Byzantine Fault Tolerance (BFT) analyses typically assume a binary state for nodes (either strictly honest or strictly malicious). However, in highly financialized cross-chain ecosystems, node behaviors are inherently profit-driven, exhibiting “bounded rationality.” To rigorously address the threats outlined in Section 3.2 mathematically, we introduce EGT to model group behavior [28,29], thereby formalizing the problem of mechanism design.

Within the EGT framework, the set of relay nodes $\mathcal{N}$ is treated as a continuously evolving population. We assume the nodes face a pure strategy space consisting of two options:

$$S = \{s_H, s_M\} \quad (2)$$

where s_H (Honest) is the honest strategy, wherein nodes strictly adhere to the cross-chain protocol, forwarding and verifying messages promptly and truthfully; s_M (Malicious) is the malicious strategy, wherein nodes forge transactions, deny services, or participate in collusion to acquire illicit gains.

Let $x(t)$ denote the proportion of nodes adopting the honest strategy s_H at time t , and $1 - x(t)$ denote the proportion adopting the malicious strategy s_M , where $x(t) \in [0, 1]$. Let U_H and U_M represent the expected payoffs for a node adopting strategies s_H and s_M , respectively. The average payoff of the population, $\bar{U}$, is defined as:

$$\bar{U} = x(t) \cdot U_H + (1 - x(t)) \cdot U_M \quad (3)$$

This study models heterogeneous cross-chain relay consensus as an N -player threshold evolutionary game based on the k -out-of- N signature mechanism. In this setting, a malicious transaction succeeds only when malicious nodes collect at least k valid-looking signatures. Therefore, the collusion success probability follows a cumulative binomial distribution rather than a linear malicious-node ratio. Let $q(t) = 1 - x(t)$ denote the malicious node ratio at time t . The threshold breach probability is formalized as:

$$P_{\text{collusion}}(q(t); N, k) = \sum_{i=k}^N \binom{N}{i} q(t)^i (1 - q(t))^{N-i} \quad (4)$$

According to Replicator Dynamics [30], the core mechanism in EGT, the evolutionary trajectory of the population’s strategy hinges on the difference between an individual’s payoff and the average payoff. When the payoff of a certain strategy exceeds the average, more nodes will tend to imitate and adopt it. The evolutionary rate of the honest node proportion $x(t)$ over time t can be expressed by the following Ordinary Differential Equation (ODE):

$$\frac{dx(t)}{dt} = x(t) \cdot (U_H - \bar{U}) = x(t)(1 - x(t))(U_H - U_M) \quad (5)$$

To reflect sustained APT behavior, the malicious payoff is further interpreted as a multi-stage discounted financial return. Across attack stages, the adversary may vary the liquidity target, bribery expenditure, stake exposure, waiting time, and exit cost. The effective malicious payoff is therefore evaluated over a prolonged horizon rather than by a single immediate gain. This extension allows the replicator dynamics model to capture strategic shifts from reputation building to collusion formation and threshold exploitation.

We also introduce a quality-game interpretation for validator interactions. Each validator chooses either a high-quality verification strategy, which incurs monitoring and communication costs but preserves trust, or a low-quality/deceptive strategy, which may

reduce short-term cost but increases slashing, demotion, and exclusion risks. A validator remains within the trusted boundary only when its evidence-backed quality score satisfies the threshold condition required for consensus participation. If the number or weight of low-quality validators approaches the k -of- N authorization threshold, the mechanism triggers SlowPath verification, additional evidence aggregation, or isolation. This quality-game boundary provides a deterministic threshold condition for separating degraded or malicious validators before the relay layer reaches a systemic failure state.

3.3.2. Problem Formulation

In traditional cross-chain bridges based on static staking, facing the temptation of massive liquidity (i.e., when V_{attack} is exceptionally large) inevitably leads to $U_M > U_H$. According to the ODE, this results in $\frac{dx(t)}{dt} < 0$, causing the system to irreversibly collapse into a state of total maliciousness ($x(t) \rightarrow 0$).

Therefore, the core optimization objective of this paper is to design a closed-loop mechanism $\mathcal{M} = \langle T(t), \Phi(T) \rangle$, encompassing a multidimensional dynamic trust evaluation function $T(t)$ and a dynamic staking constraint function $\Phi(T)$. This mechanism must satisfy the following two mathematical constraints:

(1) Security & Stability Constraint: The mechanism reconstructs the nodes' payoff matrix so that, under bounded attack value, defined threshold-signature assumptions, and selected risk-premium parameters, the honest strategy obtains a higher expected payoff than the malicious strategy. This condition drives the system toward an honest-dominant ESS in the modeled setting, rather than assuming unconditional convergence under all possible attack values.

(2) Liveness & Overhead Constraint: Predicated on satisfying the security constraint, mechanism $\mathcal{M}$ must utilize a state tiering approach to minimize destination-chain verification latency and Gas overhead for highly trusted nodes ($T_i(t) \geq \tau_{high}$).

The DTERC mechanism detailed in Section 4 is the rigorous mathematical solution and engineering implementation corresponding to the problem formulated above.

4. Methodology and System Design

Building upon the heterogeneous cross-chain system and evolutionary game models established in Section 3, this section details the proposed DTERC mechanism. The core idea of this mechanism is to construct a closed-loop, adaptive feedback system: First, it quantifies a node's historical performance through multidimensional cross-chain interaction behaviors. Second, it calculates real-time trust values by considering temporal decay and asymmetric penalty factors. Thus, it establishes a strategy for reconstructing the payoff matrix based on the EGT approach. Third, it performs hierarchical risk control relying on trust ratings, which results in achieving endogenous security in the cross-chain ecosystem.

4.1. The DTERC Framework

The DTERC mechanism is maintained jointly by smart contracts deployed on both the source and destination chains, together with an off-chain computation engine. As shown in Figure 1, the architecture consists of three basic components:

(1) Behavior Extractor (The Perception Layer): Working as the system's sensory antenna, this module extracts raw interaction data from the underlying ledgers of heterogeneous blockchains. It captures event logs produced by cross-chain gateway contracts, such as LogCrossChainRequest and LogRelayProof, through lightweight probes. When raw data are noisy because of network congestion or short-term delays, the module filters the logs and converts the unstructured records into a standardized behavioral feature vector

$\mathbf{b}_i(t)$. This vector includes basic metrics such as response latency, verification result, and online status, and these metrics are used as input for later calculations.

(2) Dynamic Trust Engine (The Computation Layer): This module performs the main trust calculation. It changes the multidimensional behavioral features into one trust value. Sliding time windows and the natural decay function are used to update historical reputation. The dynamic trust value $T_i(t)$ is then written to the on-chain Trust Registry Contract in real time. The contract stores the node reputation record, makes it publicly verifiable, and allows past trust changes to be traced.

(3) Game-Theoretic Controller (The Decision Layer): This module makes control decisions based on the trust ratings. Using the evolutionary game model, the controller tracks the change in the proportion of honest nodes in the network. When a high collusion risk is detected, it adjusts the global staking threshold D_{req} through smart contracts. This helps break the attacker's Nash equilibrium. Depending on $T_i(t)$, distribution, the manager group classifies the nodes into Whitelist, Greylist, and Blacklist automatically. Then, to conduct circuit breaking (e.g., asset freeze or node exclusion from the validator set) and differential routing allocation, it enables a paradigm shift from "passive defense" to "active isolation" in risk control.

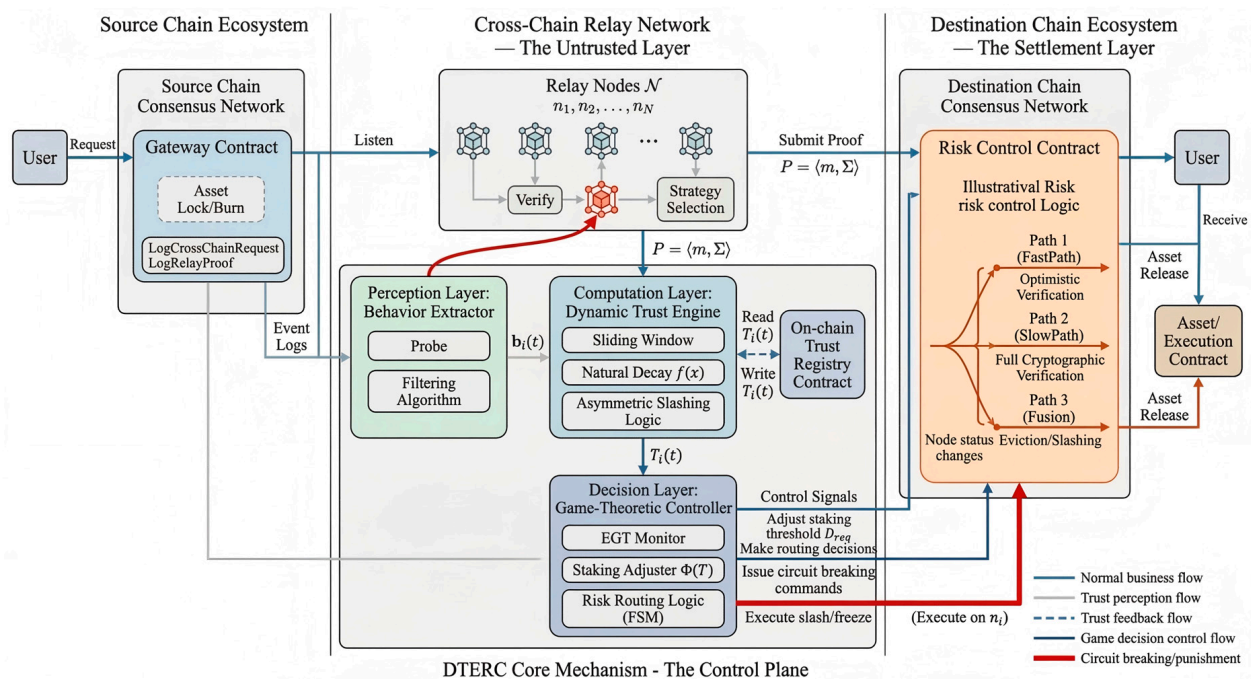


Figure 1. DTERC system architecture and workflow diagram.

4.2. Multidimensional Behavior Quantification

To fully and accurately evaluate the performance of a relay node $n_i \in \mathcal{N}$, we introduce three primary metrics on the cross-chain interaction lifecycle: Availability (response latency), Honesty (verification accuracy), and Stability (online presence).

Let t denote the current statistical time window. The behavioral vector of the node n_i over this window is formally defined as:

$$\mathbf{b}_i(t) = [b_{lat}(i, t), b_{ver}(i, t), b_{sta}(i, t)]^T \quad (6)$$

(1) Normalized Latency Index (b_{lat})

To mitigate the risk of selective relaying and lazy node attacks, this metric assesses the nodes' response times to cross-chain events. Honest nodes aim to collect and submit proofs as soon as possible, while malicious nodes tend to have high latency or timeouts. In

a globally distributed P2P network, relying on a single observer to measure the response latency is highly susceptible to geographical routing heterogeneity and transient network jitter. To eliminate subjective measurement bias, DTERC introduces a multi-observer aggregation mechanism. Let $\tau_i^{(j)}$ denote the response time of node n_i measured by the j -th independent observer. The verified latency $\bar{\tau}_i$ over window t is determined by the median of measurements reported by m randomly selected observers:

$$\bar{\tau}_i = \text{Median}\{\tau_i^{(1)}, \tau_i^{(2)}, \dots, \tau_i^{(m)}\} \quad (7)$$

To reduce the effects of heterogeneous network jitter and standardize the dimensions, we substitute $\bar{\tau}_i$ into the non-linear normalization variant of the Sigmoid function:

$$b_{lat}(i, t) = \frac{1}{1 + e^{\kappa(\bar{\tau}_i - \tau_{th})}} \quad (8)$$

Here, τ_{th} represents the benchmark latency threshold set by the system, while $\kappa > 0$ is the sensitivity coefficient. The formulation ensures that when $\bar{\tau}_i \ll \tau_{th}$, $b_{lat} \rightarrow 1$; however, a timeout triggers the score to plummet to nearly 0, forcing nodes to optimize their network response.

(2) Verification Accuracy Index (b_{ver})

Countering forged proofs and collusion fraud, this metric assesses the authenticity of verification results, being the most critical metric in the trust assessment. Let N_{total} be the total number of signatures or proofs submitted by the node within the window, N_{valid} the number of valid signatures accepted by the destination chain contract, and $N_{invalid}$ the invalid signatures were rejected or proven to be forged.

$$b_{ver}(i, t) = \frac{N_{valid}}{N_{total} + \epsilon} \times \mathbb{I}(N_{invalid} = 0) \quad (9)$$

where ϵ is a minuscule correction term (typically 10^{-6}) to prevent division by zero. The indicator function $\mathbb{I}(\cdot)$ applies the “zero-tolerance principle.” If a node has even one cryptographically confirmed forged proof, that is, $N_{invalid} > 0$, its verification score in the current cycle is set to zero. Its previous valid behavior is not counted in that cycle.

(3) Historical Online Stability Index (b_{sta})

For on-off attacks, a malicious node may repeatedly disconnect and reconnect to avoid monitoring or to reset its reputation. This metric is used to give higher scores to nodes that remain online for a long time. Let T_{online} be the cumulative effective online time of the node in the current window, and let T_{total} be the total system running time. We introduce the following switching penalty:

$$b_{sta}(i, t) = \alpha_{sta} \cdot \frac{T_{online}}{T_{total}} + (1 - \alpha_{sta}) \cdot e^{-\eta \cdot C_{switch}} \quad (10)$$

Here, $\alpha_{sta} \in (0, 1)$ controls the weight between online duration and switching frequency. C_{switch} is the number of online/offline switches during window t , and $\eta > 0$ controls the strength of the switching penalty. A larger η gives a stronger penalty to frequent switching. This equation means that a stable node should keep a high online rate and also avoid frequent connection changes. In this way, the system can identify nodes that repeatedly leave and rejoin the network to avoid detection.

Combining the three dimensions, the current comprehensive behavioral score $S_i(t)$ for node n_i at time window t is defined as:

$$S_i(t) = w_1 b_{lat} + w_2 b_{ver} + w_3 b_{sta}, \text{ s.t. } \sum_{k=1}^3 w_k = 1 \quad (11)$$

The adaptive weights w_1, w_2, w_3 can be dynamically adjusted according to the specific security requirements of the cross-chain scenario (e.g., w_2 can be elevated in asset-security-first applications).

4.3. Dynamic Trust Calculation Based on Temporal Decay and Asymmetric Penalty

Conventional reputation accumulation models are vulnerable to concealed on-off attacks and whitewashing attacks in cross-chain environments. To address these threats, DTERC uses a time-dependent and asymmetric trust-update mechanism: trust should accumulate gradually through repeated compliant behavior, whereas confirmed malicious evidence should trigger rapid trust degradation through a circuit-breaker rule.

4.3.1. Temporal Decay Model of Trust

In heterogeneous cross-chain environments, node behavior is time-dependent. A dormant node that has never been verified for a long time should not keep its high historical reputation forever; otherwise, attackers may exploit its old reputation and launch an attack after a dormant period. To address this, we introduce the physical half-life to develop a temporal decay model for trust.

Let Δt be the time elapsed between the current update window and the last active window. The natural decay function for historical trust, $D_{hist}(\Delta t)$, is defined as:

$$D_{hist}(\Delta t) = T_i(t - \Delta t) \cdot e^{-\lambda \Delta t} \quad (12)$$

where $\lambda > 0$ is the decay constant, dictating the rate at which reputation dissipates over time.

We define the trust half-life as $\tau_{1/2} = \ln(2)/\lambda$. If a node has no cross-chain interaction for a period of $\tau_{1/2}$, that is, $\Delta t = \tau_{1/2}$, its trust value is reduced by half. This setting makes inactive nodes lose trust over time. Honest nodes, therefore, need to remain online and take part in verification regularly. For hidden attackers, keeping a high trust score without real activity becomes more difficult [31].

4.3.2. Adaptive Recursive Update Algorithm

Using the current behavioral score $S_i(t)$ defined in Section 4.2 and the temporal decay model, the node trust value is updated recursively. The smoothing factor preserves part of the historical reputation while incorporating the node's current behavior, thereby reducing sensitivity to short-term measurement noise.

To limit the benefit of adversarial behavior, we introduce an asymmetric slashing mechanism. The trust update is not treated as a simple linear weighted average; instead, it depends on whether verifiable malicious behavior has been detected, as indicated by $N_{invalid}$.

Theorem 1. *Dynamic Trust Update Equation.*

The global trust value $T_i(t) \in [0, 1]$ of node n_i at time t is determined by two parts: the decayed historical reputation and the current behavioral score. $T_i(t)$ is defined by the following piecewise function:

$$T_i(t) = \begin{cases} \mu \cdot T_i(t - \Delta t) \cdot e^{-\lambda \Delta t} + (1 - \mu) \cdot S_i(t) & \text{if } N_{invalid} = 0 \\ 0 & \text{if } N_{invalid} > 0 \end{cases} \quad (13)$$

Here, $\mu \in (0, 1)$ is the Trust Inertia Factor. A higher μ value, for example, 0.8, means that the system gives more weight to long-term behavior. In this case, trust grows more slowly and is less affected by short network fluctuations. A lower μ value makes the system more sensitive to current behavior. We use a smaller μ during system initialization to make convergence faster, and a larger μ during stable operation to improve robustness.

The central design principle of this equation is asymmetric slashing. Once malicious behavior is confirmed by threshold-signed evidence and the challenge procedure, the trust value is reset to the baseline. Thus, trust is accumulated gradually but can be lost immediately after confirmed malicious conduct, which reduces the incentive for a node to accumulate reputation and spend it on a single high-value attack.

4.3.3. Algorithm Implementation

Algorithm 1 implements the above trust-update logic. The implementation does not rely on a single trusted off-chain oracle. Instead, multiple independent observers collect behavior evidence, aggregate continuous signals such as latency through robust statistics, and submit threshold-signed reports to the Trust Registry Contract. Security-critical evidence, including invalid signatures, forged proofs, and conflicting cross-chain messages, is audited through threshold voting and a challenge window before slashing or blacklisting is finalized.

Algorithm 1: Dynamic Trust Evaluation Algorithm

Input:

Nodes set $\mathcal{N}$, Time window t , Logs of interactions $\mathcal{L}_t$, Previous trust vector $\mathbf{T}(t - \Delta t)$, Parameters $\lambda, \mu, \tau_{th}, \kappa$.

Output:

Updated trust vector $\mathbf{T}(t)$

- 1: **for** each node $n_i \in \mathcal{N}$ **do**
 - 2: Step 1: Extract Features
 - 3: Calculate $\bar{\tau}_i, N_{valid}, N_{invalid}, C_{switch}$ from $\mathcal{L}_t$;
 - 4: Step 2: Check Malicious Activity (Pre-screening)
 - 5: **if** $N_{invalid} > 0$ **then**
 - 6: $T_i(t) \leftarrow 0$ // Trigger Asymmetric Slashing;
 - 7: Broadcast RiskAlert (n_i) to Target Smart Contract;
 - 8: **continue**;
 - 9: **end if**
 - 10: Step 3: Quantify Behavior
 - 11: Calculate $b_{lat}, b_{ver}, b_{sta}$ using Equations (1)–(3);
 - 12: $S_i(t) \leftarrow w_1 b_{lat} + w_2 b_{ver} + w_3 b_{sta}$;
 - 13: Step 4: Apply Time Decay and Update
 - 14: $\Delta t \leftarrow t - t_{last_active}(n_i)$;
-

```

15:  $T_{decay} \leftarrow T_i(t-1) \cdot e^{-\lambda \Delta t}$ ;
16:  $T_i(t) \leftarrow \mu \cdot T_{decay} + (1-\mu) \cdot S_i(t)$ ;
17: end for
18: return  $\mathbf{T}(t)$ 

```

Compared to traditional trust models such as Beta distribution or simple weighted averages, the proposed algorithm has the following advantages:

(1) Temporal Reputation Decay: The temporal decay factor $e^{-\lambda \Delta t}$ is used to model the natural dissipation of inactive reputation. This design prevents dormant or zombie nodes from retaining high trust values indefinitely and reduces the feasibility of reputation-hoarding attacks.

(2) Asymmetric Design from a Game-Theoretic Perspective: Trust accumulation follows a gradual process $O(t)$, whereas confirmed malicious behavior causes abrupt trust degradation $O(1)$. This asymmetry makes short-term malicious gains U_M less attractive than long-term honest participation U_H , thereby providing the behavioral basis for the ESS analysis in Section 4.4.

4.4. Evolutionary Game Stability Proof and Payoff Reconstruction

Referring to the system model in Section 3.3, the objective is to drive the relay population toward an honest-dominant equilibrium under bounded attack value, defined threshold-signature assumptions, and selected risk-premium parameters. Achieving this objective requires reconstructing the node payoff matrix so that the expected payoff of the honest strategy exceeds that of the malicious strategy in the modeled setting [32], namely $U_H(t) > U_M(t)$.

4.4.1. Payoff Matrix Reconstruction Based on Trust

The rewards and penalties in traditional cross-chain models are mostly static, which does not consider the volatility of attack costs. They fail to adapt to dynamic changes in attack cost. This paper provides trust-tier-based dynamic payoff functions.

(1) Dynamic Payoff Function for the Honest Strategy ($U_H(t)$)

An honest node's payoff comprises a base verification reward (Gas fee dividends) and a reputation incentive, minus computational and communication costs.

$$U_H(i, t) = \underbrace{R_{base} \cdot \frac{T_i(t)}{\sum_{j \in \mathcal{N}} T_j(t)}}_{\text{Trust-Weighted Reward}} + \underbrace{\beta_{inc} \cdot \ln(1 + T_i(t))}_{\text{Reputation Incentive}} - C_{ver} \quad (14)$$

where R_{base} is the total cross-chain transaction fee pool within the current block; β_{inc} is the reputation incentive coefficient. The logarithmic function $\ln(1 + T_i(t))$ is introduced to generate “diminishing marginal returns,” encouraging low-to-medium reputation nodes to rapidly accumulate credit while preventing high-reputation nodes from monopolizing excessive surplus returns, thereby sustaining system decentralization. C_{ver} denotes the computational and network costs of executing verification algorithms and signature aggregation.

(2) Dynamic Payoff Function for the Malicious Strategy ($U_M(t)$)

The payoff for a malicious node is the gain from the attack minus the total penalty that the node receives:

$$U_M(i, t) = P_{collusion}(q(t); N, k) \cdot V_{attack} - \mathcal{L}_{total}(i, t) \quad (15)$$

where V_{attack} is the total value of illicit assets the attacker attempts to acquire via forged transactions or collusion double-spending. Crucially, as derived in the N -player threshold game model, the potential attack windfall is strictly weighted by the real-time threshold breach probability $P_{collusion}(q(t); N, k)$. $\mathcal{L}_{total}$ represents the comprehensive loss function post-attack, acting as the core constraint variable in our mechanism design.

4.4.2. Dynamic Staking Constraint Mechanism

To break down the “high return from the attack” Nash equilibrium, a dynamic staking constraint is created. Rather than requiring all nodes to deposit the same amount of stake, the system adjusts the entry threshold according to $T_i(t)$.

Theorem 2. *Dynamic Staking Function $\Phi(T_i)$.*

The minimum staking requirement $D_{req}(i, t)$ for node n_i to enter the current consensus round varies according to its trust value as follows:

$$D_{req}(i, t) = D_{base} \cdot (1 + \gamma_{risk} \cdot e^{-\xi \cdot T_i(t)}) \quad (16)$$

Here, D_{base} is the system’s baseline staking amount; γ_{risk} is the risk premium coefficient, and ξ denotes trust sensitivity. If trust is low, the node must provide a significant risk premium to be able to participate in the verification (expensive constraint), while if the trust is high, the staking threshold is much lower (low-cost incentive).

Therefore, the total loss function $\mathcal{L}_{total}$ for the malicious strategy is:

$$\mathcal{L}_{total}(i, t) = \underbrace{D_{req}(i, t)}_{\text{Immediate Slashing}} + \underbrace{\sum_{\tau=t+1}^{\infty} \delta^{\tau-t} \cdot U_H(i, \tau)}_{\text{Opportunity Cost}} \quad (17)$$

Immediate Slashing: Upon malicious execution, the staked amount D_{req} is entirely slashed. Notably, for low-trust potential attackers, their D_{req} is exceedingly high, directly neutralizing V_{attack} .

Opportunity Cost: $\delta \in (0, 1)$ is the discount factor. After an attack, the node’s trust value T_i becomes zero, and the node is moved to the blacklist. Therefore, it loses the honest rewards that it could have earned in later rounds. For high-trust nodes, the immediate staking amount D_{req} may be low, but the loss of future rewards is high. This makes betrayal unattractive for such nodes.

4.4.3. Proof of Evolutionary Stability

Theorem 3. *Under bounded attack value $T_i(t)$, defined threshold-signature assumptions, and selected risk-premium parameters, the integration of dynamic trust and the dynamic staking function $\Phi(T_i)$ drives the relay population toward an honest-dominant ESS in the modeled setting.*

Proof of Theorem 3. Substituting the reconstructed payoffs into the Replicator Dynamics equation gives

$$\frac{dx(t)}{dt} = x(t)(1 - x(t)) [U_H(i, t) - (V_{attack} - D_{req}(i, t) - \text{OppCost})] \quad (18)$$

For $\frac{dx(t)}{dt} > 0$ to hold for any $x(t) \in (0, 1)$, it is enough to show that

$$D_{req}(i, t) + \text{OppCost} + U_H(i, t) > V_{attack} \quad (19)$$

(1) Low-trust nodes ($T_i \rightarrow 0$): According to the dynamic staking function, a node with low trust must provide a much higher stake. By choosing a proper value of γ_{risk} , the system can make D_{req} larger than V_{attack} . In this case, the inequality in Equation (17) holds.

(2) High-trust nodes ($T_i \rightarrow 1$): For a high-trust node, D_{req} approaches D_{base} . However, such a node also has more future honest rewards to lose. Once it attacks, its trust is reset, and these future rewards become the opportunity cost. Since the value of a cross-chain attack is bounded by the liquidity of the bridge, the opportunity cost and the remaining terms in Equation (17) can still make the attack unprofitable.

Therefore, under the DTERC mechanism and the specified parameter conditions, the honest strategy obtains a higher expected payoff than the malicious strategy. Thus, $\frac{dx(t)}{dt} > 0$, the relay population tends to move toward an honest-dominant equilibrium in the modeled setting. $\square$

4.4.4. Game-Theoretic Risk Control Algorithm Implementation

The game-theoretic logic is engineered through Algorithm 2. Executed by the Game-Theoretic Controller, this algorithm dynamically computes the staking threshold per round and monitors the network's convergence state.

Algorithm 2: Game-Theoretic Risk Control Algorithm

Input:

Current trust vector $\mathbf{T}(t)$, Network average trust $\bar{\tau}_{net}$, Base stake D_{base} , Risk parameters γ_{risk}, ξ

Output:

Staking requirements vector $\mathbf{D}_{req}(t)$, System State $\mathcal{S}_{sys}$

```

1: Initialize:  $\mathcal{S}_{sys} \leftarrow \text{STABLE}$ 
2: for each node  $n_i \in \mathcal{N}$  do
3:   Step 1: Calculate Dynamic Staking Threshold;
4:   Compute risk factor  $\rho_i = e^{-\xi \cdot T_i(t)}$ ;
5:    $D_{req}(i, t) \leftarrow D_{base} \cdot (1 + \gamma_{risk} \cdot \rho_i)$ ;
6:   Update smart contract:  $\text{setStakingRequirement}(n_i, D_{req}(i, t))$ ;
7:   Step 2: Stability Check (ESS Convergence)
8:   Compute Expected Honest Payoff  $U_H(i)$  using Equation (Trust-Weighted Reward);
9:   Compute Potential Attack Gain  $V_{attack}$  (based on current TX pool);
10:  if  $U_H(i) + D_{req}(i, t) < V_{attack}$  then
11:    //Warning: Nash Equilibrium Broken
12:     $\mathcal{S}_{sys} \leftarrow \text{UNSTABLE}$ ;
13:  Increase global risk parameter:  $\gamma_{risk} \leftarrow \gamma_{risk} \cdot 1.5$ ;
14:  Trigger EmergencyPause() if critical;
15: end if
16: end for
17: return  $\mathbf{D}_{req}(t), \mathcal{S}_{sys}$ 

```

4.5. Hierarchical Risk Control Strategy

To balance the security and liveness of the cross-chain system while implementing differential management for nodes of varying reputations, we design a hierarchical risk control architecture based on a Finite State Machine (FSM). This architecture divides relay nodes into three separate security states according to their real-time trust values $T_i(t)$. It then controls their privileges and routing permissions dynamically.

4.5.1. Three-Level Trust State Classification

The system establishes two critical trust thresholds: the admission threshold τ_{low} and the high-reputation threshold τ_{high} (where $0 < \tau_{low} < \tau_{high} < 1$). Based on the interval distribution of $T_i(t)$, nodes are automatically classified into the following three tiers, with smart contracts executing differential resource allocation strategies:

(1) Trusted State (Whitelist, $T_i(t) \geq \tau_{high}$)

Core nodes reserved for the Trusted State deliver high performance over many epochs. They require the lowest staking threshold ($D_{req} = D_{base}$) and are assigned high routing weights for cross-chain verification. These nodes are preferentially chosen to serve as a Leader node in aggregate signatures or a participant in threshold signatures, thereby maximizing the fee revenues (maximizing U_H). Proofs submitted by these nodes can trigger the Optimistic Verification logic within the destination chain's smart contract, thereby entering the FastPath and minimizing the cross-chain verification latency. It is imperative to clarify that the FastPath mechanism does not equate to a Single Point of Failure (SPOF) dictatorial channel. Instead, it is safeguarded by a two-tier security net inspired by Optimistic Verification principles. First, it requires a micro-threshold consensus strictly within the Whitelist committee (e.g., 2/3 of Whitelist nodes) to prevent isolated key leaks from forging valid submissions. Second, for candidates that pass the FastPath micro-threshold, DTERC further introduces a challenge window before final settlement. During this period, any inconsistent state proof, invalid signature, forged event log, or conflicting cross-chain message can trigger a challenge. Once a challenge is verified, the transaction is rolled back or redirected to SlowPath, and the involved malicious whitelisted nodes are immediately downgraded or blacklisted through the one-strike veto rule.

Therefore, FastPath improves efficiency by reducing redundant verification and prioritizing low-risk transactions under normal conditions, but it does not replace the threshold-signature-based security model. Under adversarial conditions, FastPath automatically sacrifices part of its performance advantage by triggering challenge-based fallback or SlowPath verification, thereby preserving the decentralized security boundary of the relay network.

(2) Probationary State (Greylist, $\tau_{low} < T_i(t) < \tau_{high}$)

This class includes first-time nodes or nodes whose ratings drop because of network jitter. They must pay a high-risk premium and have only basic message forwarding privileges. They are strictly prohibited from serving as consensus committee Leaders and cannot verify high-value cross-chain assets. Any proofs submitted by the Greylist nodes will be subjected to strict cryptographic re-verification via random node sampling (Challenge Mechanism).

(3) Fused State (Blacklist, $T_i(t) \leq \tau_{low}$)

The circuit breaker targets “malicious nodes” that have triggered the circuit breaker or “zombie nodes” whose trust has naturally decayed beyond the elimination line. The smart contract invokes the freeze() interface and locks their entire staked assets D_{req} outside the active Validator Set. The system rejects incoming messages signed by them. A RiskAlert event is emitted and synchronized to all associated destination chains, ensuring that the network can prevent further attacks on other related chains.

4.5.2. State Transition and the “One-Strike Veto” Circuit Breaker

The transitions between these three states are developed from Algorithm 1, which updates trust values dynamically. The transition process is shown in Figure 2.

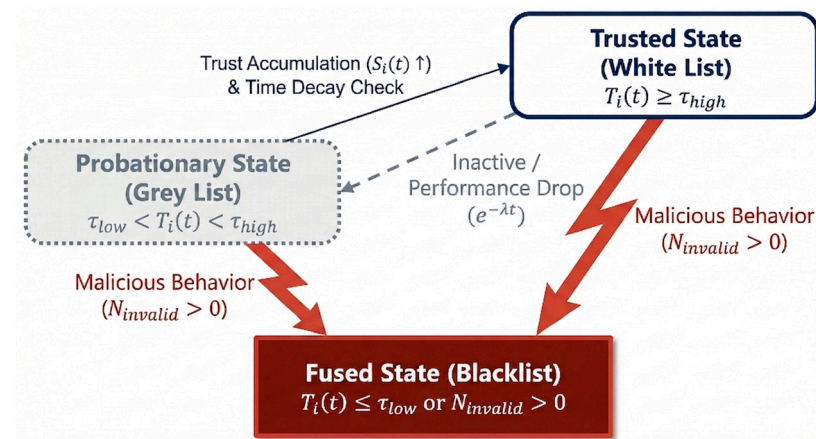


Figure 2. State transition diagram of the hierarchical risk control mechanism based on dynamic trust evaluation.

Progressive Promotion: Greylist nodes can gradually upgrade from the Greylist to the Whitelist by consistently accumulating $S_i(t)$ against the temporal decay $e^{-\lambda \Delta t}$. This constitutes a linear accumulation process over time.

Precipitous Demotion:

(1) **Natural Demotion:** The node will naturally fall to Greylist if it remains inactive for an extended period or exhibits lower service quality, causing $T_i < \tau_{high}$.

(2) **Circuit Breaker Demotion:** The mechanism aims to prevent on–off attacks with a “One-Strike Veto” mechanism. If $N_{invalid} > 0$ (which indicates a single cryptographically confirmed malicious act), regardless of the node’s previous standing, this mechanism will directly trigger a circuit breaker interrupt. The node is immediately moved to the Blacklist, and its collateral is fully slashed.

This asymmetric design reflects the game-theoretic constraints proven in Section 4.4 and ensures that the attacker’s long-term expected loss outweighs any illicit gain.

5. Security Analysis

Based on the threat model established in Section 3 and the DTERC mechanism detailed in Section 4, this section analyzes the security properties of DTERC under the stated assumptions and parameter conditions. We present several theorems corresponding to typical adversarial behaviors and examine them using mathematical derivation.

5.1. Security Against On–Off/Traitor Attacks

Theorem 4. For any rational on–off attacker $\mathcal{A}_{on-off}$, under the asymmetric penalty and opportunity cost constraints of DTERC, if the system satisfies $\sum_{\tau=t+1}^{\infty} \delta^{\tau-t} \cdot U_H(i, \tau) > V_{attack} - D_{req}(i, t)$; the attacker cannot obtain a positive net payoff through the strategy of “long-term honest accumulation followed by sudden maliciousness.”

Proof of Theorem 4. Assume an attacker node n_i adopts the honest strategy s_H during the interval $[0, T_{switch})$, accumulating a high trust value $T_i(T_{switch}^-) \rightarrow 1$. At exactly $t = T_{switch}$, it abruptly switches to the malicious strategy s_M to launch a double-spending attack, attempting to seize illicit gain V_{attack} . According to Definition 1 (Dynamic Trust Update) in Section 4.3, the moment the oracle detects the forged proof ($N_{invalid} > 0$) at T_{switch} , the node’s trust value triggers the one-strike veto circuit breaker, instantly plunging to the baseline: $T_i(T_{switch}) = 0$. To avoid relying on a fully trusted external oracle, DTERC treats oracle reports as noisy and potentially adversarial evidence rather than as deterministic ground truth. DTERC adopts a multi-oracle evidence aggregation layer, where reports

from multiple independent oracle nodes are aggregated through robust statistical and threshold-based mechanisms. Specifically, continuous trust-related signals, such as latency, response time, and availability, are aggregated using robust estimators. Discrete security events, such as invalid signatures, forged cross-chain messages, conflicting state proofs, and omission of malicious behavior, are determined through threshold voting. This design reduces the impact of erroneous, delayed, or malicious oracle reports and prevents a single compromised oracle from manipulating the trust update process.

At this point, the attacker's total expected payoff $\mathbb{E}[\mathcal{A}]$ at and after T_{switch} is:

$$\mathbb{E}[\mathcal{A}] = V_{attack} - \mathcal{L}_{total}(i, T_{switch}) \quad (20)$$

Per Equation (15), the comprehensive loss $\mathcal{L}_{total}$ comprises immediate slashing and future opportunity cost:

$$L_{total} = \underbrace{D_{req}(i, T_{switch})}_{\text{Slashing}} + \underbrace{\sum_{\tau=T_{switch}+1}^{\infty} \delta^{\tau-T_{switch}} \cdot U_H(i, \tau)}_{\text{Opportunity Cost}} \quad (21)$$

Because the trust value is zeroed and the node is demoted to the Blacklist ($T_i \leq \tau_{low}$), during the arduous “trust climbing phase” constrained by the inertia factor μ , the node forfeits all honest verification rewards, meaning $U_H(i, \tau) \approx 0$ for the near future τ . Provided the system parameters enforce:

$$V_{attack} < D_{req}(i, T_{switch}) + \sum_{\tau=T_{switch}+1}^{\infty} \delta^{\tau-T_{switch}} \cdot U_H(i, \tau) \quad (22)$$

Then $\mathbb{E}[\mathcal{A}] < 0$, indicating that a rational attacker's net payoff for launching an on-off attack is negative under the specified parameter settings. Consequently, the attacker has no incentive to choose the strategy s_M . $\square$

5.2. Security Against Rational Collusion

Theorem 5. Suppose an attacker attempts to manipulate c relay nodes (satisfying the threshold requirement $c \geq k$) for collusion. Under the dynamic staking constraints of DTERC, the collusion cost function $C_{collusion}$ is a convex function regarding the participating nodes' trust values T_i . Furthermore, for any arbitrary combination of nodes, $C_{collusion} > V_{attack}$ holds under the specified parameter settings, thereby dismantling rational collusion motives.

Proof of Theorem 5. Let the colluding node set be $\mathcal{C} = \{n_{c_1}, \dots, n_{c_c}\} \subseteq \mathcal{N}$. If the forged proof submitted by these nodes is detected, for example, through a challenge period or proof verification, all colluding nodes lose their staked capital. Therefore, the direct capital cost of the attacker is:

$$C_{collusion} = \sum_{n_i \in \mathcal{C}} D_{req}(i, t) \quad (23)$$

According to Equation (14), we consider two typical cases:

(1) Low-trust node collusion (i.e., Sybil node collusion): If all colluding nodes have low trust, that is, $\forall n_i \in \mathcal{C}, T_i(t) \rightarrow 0$, then each node must provide a high staking amount. In this case, the total cost is approximately:

$$C_{collusion} \approx c \cdot D_{base}(1 + \gamma_{risk}) \quad (24)$$

By setting the risk premium coefficient as $\gamma_{risk} > \frac{V_{attack}}{c \cdot D_{base}} - 1$ the system can ensure that $C_{collusion} > V_{attack}$.

(2) High-trust node collusion: If $\exists n_i \in \mathcal{C}, T_i(t) \rightarrow 1$.

If some colluding nodes have high trust, that is, $\exists n_i \in \mathcal{C}, T_i(t) \rightarrow 1$, their immediate staking requirement may be close to D_{base} . However, these nodes also have higher future rewards because of their trust weights and reputation incentives. Once they are detected as malicious, they lose not only the current stake but also future honest returns. The attacker's total loss can be written as:

$$\mathcal{L}_{total} = \sum_{n_i \in \mathcal{C}} (D_{req}(i, t) + \text{OppCost}_i) \quad (25)$$

Because OppCost_i exhibits super-linear growth with respect to $T_i(t)$ (due to the dual effect of dividend weights and reputation incentives), the total loss $\mathcal{L}_{total}$ exceeds the singular attack gain V_{attack} .

Overall, low-trust nodes face high staking costs, while high-trust nodes face high opportunity costs. Under these two constraints, the total expected loss of collusion is greater than the possible attack gain. Hence, rational collusion has a negative expected payoff, and the attacker will not choose this strategy. $\square$

5.3. Security Against Selective Relaying and Bad-Mouthing

Theorem 6. Under the DTERC multidimensional scoring framework, a node n_{mal} adopting “selective relaying” or “bad-mouthing” strategies faces a strict convergence ceiling for its trust value that is lower than that of an honest node n_{hon} . Correspondingly, its long-term expected payoff U_M^{mal} is strictly bounded below U_H^{hon} .

Proof of Theorem 6. (1) Against Selective Relaying: Suppose n_{mal} deliberately introduces extra latency $\delta_t > 0$ to specific cross-chain requests. Its average response time becomes $\bar{\tau}_{mal} = \bar{\tau}_{hon} + \delta_t$. Substituting this into Equation (6) yields:

$$b_{lat}(mal, t) = \frac{1}{1 + e^{\kappa(\bar{\tau}_{hon} + \delta_t - \tau_{th})}} < b_{lat}(hon, t) \quad (26)$$

Consequently, its current comprehensive score is $S_{mal}(t) < S_{hon}(t)$. Substituting into the trust update Equation (11):

$$T_{mal}(t) = \mu T_{mal}(t - \Delta t) e^{-\lambda \Delta t} + (1 - \mu) S_{mal}(t) \quad (27)$$

Whether in the initial or steady state, a persistently lower S_{mal} depresses the sequence's limit, ensuring $T_{mal}(t) < T_{hon}(t)$ invariably.

(2) Against Bad-mouthing: If n_{mal} intentionally refuses to provide signatures for n_{hon} during the aggregate signature phase (attempting to degrade the honest node's performance evaluation), this behavior is recorded in off-chain event logs as a “verification absence” for n_{mal} . While this may cause the current consensus round to fail, the traceback mechanism attributes the absence solely to the non-signing n_{mal} , degrading its verification index b_{ver} . As long as n_{hon} performs normally in other responses, its long-term score remains unblemished. According to Equation (12), routing weights and incentive payoffs are directly proportional to the trust value $T_i(t)$; therefore, this inevitably yields $U_M^{mal} < U_H^{hon}$.

Overall, maliciously degrading Quality of Service (QoS) or defaming competitors constitutes a strictly dominated strategy in the game. $\square$

5.4. Security Against Sybil Attacks

Theorem 7. Under the dynamic staking rule and the stated parameter settings, DTERC increases the economic cost of Sybil identities. Specifically, an attacker forging M fake identities obtains a total payoff U_{sybil} that does not exceed the total committed cost C_{entry} .

Proof of Theorem 7. Suppose an attacker generates M Sybil identities $\{id_1, \dots, id_M\}$.

Admission Cost Defense: All new identities lack historical interaction records, initially with $T_{init} \approx 0$. Per the dynamic staking rule, each identity must deposit a high-risk margin:

$$C_{entry} = \sum_{j=1}^M D_{base}(1 + \gamma_{risk}e^{-\xi \cdot 0}) = M \cdot D_{base}(1 + \gamma_{risk}) \quad (28)$$

Payoff Isolation Defense: Because their initial trust is negligible, Sybil nodes are relegated to the “Probationary Greylist.” They are barred from processing large-value funds, and their routing weight for cross-chain task allocation is $W_{route} \propto T_{init} \approx 0$. During the protracted trust climbing phase $[0, \tau_{climb}]$, $U_{sybil} \approx 0$.

Temporal Decay Defense: If Sybil nodes attempt to increase T_i through honest behavior, trust accumulation takes a long time with inertia μ and physical time. During this period, their massive, staked capital C_{entry} is locked, and the time value of money (liquidity attrition) further magnifies their losses. If the attacker launches an attack before trust increases, high staking costs make the attack unprofitable. Conversely, if the attacker gains trust honestly, they have game-theoretically become honest nodes in the system, eliminating the Sybil threat. Therefore, $U_{sybil} - C_{entry} < 0$. $\square$

Table 2 summarizes the theorems’ threat models and core defenses, clearly illustrating the DTERC mechanism’s security characteristics.

Table 2. Summary of Security Features of the DTERC Mechanism against Typical Cross-Chain Threats.

Threat Type	Attacker Motivation and Characteristics	Security Theorem	Core Defense Component in DTERC
Systemic Rational Deviation	Rational nodes dynamically migrating towards malicious strategies driven by higher expected payoffs, threatening systemic collapse.	Theorem 3	Payoff matrix reconstruction via threshold-aware EGT; honest-dominant convergence under bounded assumptions.
On-off/Traitor Attack	Long-term honest latency followed by sudden malicious behavior for windfall profits.	Theorem 4	Asymmetric “one-vote veto” circuit breaker; Opportunity cost constraint.
Rational Collusion	Controlling more than the statutory consensus threshold k nodes to commit fraud.	Theorem 5	Dynamic risk premium staking function $D_{req}(T_i)$.
Selective Relaying	Maliciously introducing latency or refusing to sign to suppress competitors.	Theorem 6	Multidimensional QoS quantitative normalization; Time decay iteration.
Sybil Attack	Low-cost replication of fake identities to manipulate the network consensus.	Theorem 7	Punitive entry capital threshold for zero-trust nodes.

6. Experimental Verification and Performance Analysis

To evaluate the proposed DTERC mechanism’s effectiveness, resilience, and prototype-level viability, this section conducts a series of experiments. The evaluation is based on a multi-agent evolutionary simulation environment and an Ethereum/Hyperledger

Fabric prototype. The four key Research Questions (RQs) are framed for the experiments, as follows:

RQ1 (Trust Convergence and Agility): Does DTERC accurately profile trust and trigger circuit breaking reliably in the presence of complex malicious behaviors, especially concealed on–off attacks?

RQ2 (Evolutionary Game and Collusion Resistance): Can the network under dynamic staking move toward an honest-dominant ESS under bounded attack-value and parameter conditions when facing rational collusion at varying scales?

RQ3 (Parameter Sensitivity and Robustness): Does the mechanism maintain robustness to noise and evolutionary deterrence under the hyperparameter variations and extreme network jitter?

RQ4 (System Overhead and Prototype-Level Feasibility): Do the end-to-end cross-chain latency and on-chain Gas cost remain within a reasonable range after trust is continuously computed and state updated?

6.1. Experimental Setup and Parameters

Because the underlying relay networks of active cross-chain mainnets (e.g., LayerZero, Wormhole) run in closed environments that prevent active collusion or on–off attacks, we use an off-chain multi-agent simulation and an on-chain smart contract prototype, following an “off-chain multi-agent simulation + on-chain smart contract prototype” design. The server includes an Intel Core i9-13900K CPU and 64 GB of memory running on Ubuntu 22.04 LTS.

(1) Evolutionary Game Simulation Layer: Using the Python 3.9 multi-agent modeling environment Mesa, we construct a heterogeneous network of $N = 100$ cross-chain relay nodes. The cross-chain transaction request’s arrival rate is Poisson-distributed with $\lambda_{\text{req}} = 20$ tx/s. This layer simulates multidimensional interaction behaviors, creates time-series feature data, and tracks the game convergence states of different trust mechanisms.

(2) Performance Evaluation Prototype Layer: To evaluate engineering viability and Gas overhead, we use a local EVM-compatible testnet powered by the Hardhat framework for the source chain and the Hyperledger Fabric v2.4 for the destination chain. Trust computation, dynamic staking, and slashing are implemented in Solidity 0.8.19, and Ethers.js orchestrates state synchronization to the off-chain solver.

It is especially important to mention that the system’s core parameters’ initialization was not chosen arbitrarily; rather, it follows real historical interaction data between Ethereum and Polygon from real cross-chain bridges, and the theoretical boundary conditions learned in the previous section. The detailed system initialization parameters can be found in Table 3.

For comparative analysis, we selected three baseline mechanisms as experimental control groups, spanning industrial standards, classical theories, and state-of-the-art academic achievements:

Static-PoS (Static Proof of Stake): Represents the industrial standard of current mainstream cross-chain bridges (e.g., AnySwap, Wormhole). All nodes paying a fixed D_{base} acquire identical routing weights. The system lacks continuous evaluation of historical behavior, relying exclusively on post-event slashing only after cryptographically provable malicious facts transpire.

Table 3. Core Simulation Parameters and Grounding Rationale of the DTERC System.

Parameter	Description	Value	Grounding Rationale & Physical Significance
N	Total population of relay nodes	100	Simulates the typical node capacity of medium-to-large decentralized cross-chain oracle networks.
k	Statutory threshold for aggregate signatures	67	Adheres to the conventional 2/3 honest majority assumption in BFT consensus.
τ_{th}	Benchmark latency tolerance threshold	200 ms	Anchored to the average P2P broadcast network latency between Ethereum mainnet nodes.
κ	Latency sensitivity coefficient	0.1	Controls the steepness of the Sigmoid function decay; determined via pre-experimental tuning.
μ	Trust the inertia factor	0.8	Endows the system with robust “historical memory” to filter rating fluctuations caused by episodic network jitter.
λ	Trust decay constant	0.05	Establishes the expected “trust half-life”; chronically dormant nodes degrade to the Greylist at this rate.
τ_{high}	Whitelist/Trusted state threshold	0.85	Ensures only nodes sustaining protracted high online rates and verification accuracy gain entry.
τ_{low}	Blacklist/Elimination threshold	0.4	Nodes falling below this metric are deemed “zombie nodes,” triggering asset freezing and network eviction.
γ_{risk}	Risk premium coefficient	2.5	Guarantees that when low-trust nodes participate in collusion, their staking cost scales exponentially, satisfying $U_M \ll 0$.

Beta-Reputation [33]: Represents the heuristic trust evaluation academic baseline widely adopted in distributed networks. Leveraging the conjugate prior property of the Beta distribution, it calculates expected trust $\mathbb{E}(T) = \frac{\alpha}{\alpha+\beta}$ by accumulating positive (α) and negative (β) interactions. Notably, it lacks temporal decay and an asymmetric “one-strike veto” penalty design.

DDRV (Decentralized Dynamic Reputation Value) [17]: Represents a recent trust model specifically for cross-chain settings. Employing a sliding window and multidimensional QoS aggregation, it adapts relatively rapidly to variations in the network. However, its penalty mechanism uses linear decline and lacks deep coupling with base staking capital.

6.2. Trust Convergence and Resistance to On–Off Attacks

This section examines the accuracy of the DTERC mechanism in tracing node behaviors, emphasizing its fast response and non-repudiation in the face of the highly damaging “on–off attack” (alternatively, the traitor attack).

6.2.1. Experimental Scenario and Metrics

In the multi-agent simulation setting, an APT instance mimics tracking the trust evolution of a given on–off attacker.

Behavioral Setting: The node is fully protocol-compliant within the t time window, exhibiting low latency and 100% verification accuracy. Yet, precisely at $t = 500$, it suddenly launches a forged proof attack at a cross-chain transaction of exceptional value (i.e., generating $N_{invalid} > 0$). Therefore, it remains to be verified to obtain more illicit profit $t \in (500, 600]$.

Evaluation Metrics: Steady-state Trust, Isolation Latency (defined as the time steps before the trust value descends below the elimination threshold $\tau_{low} = 0.4$), and the False Negative Rate (FNR) within the Malicious Residual Window.

6.2.2. Dynamic Trust Evolution Analysis

Figure 3 shows the trust score evolutionary curves of the on–off attacker across varying evaluation mechanisms.

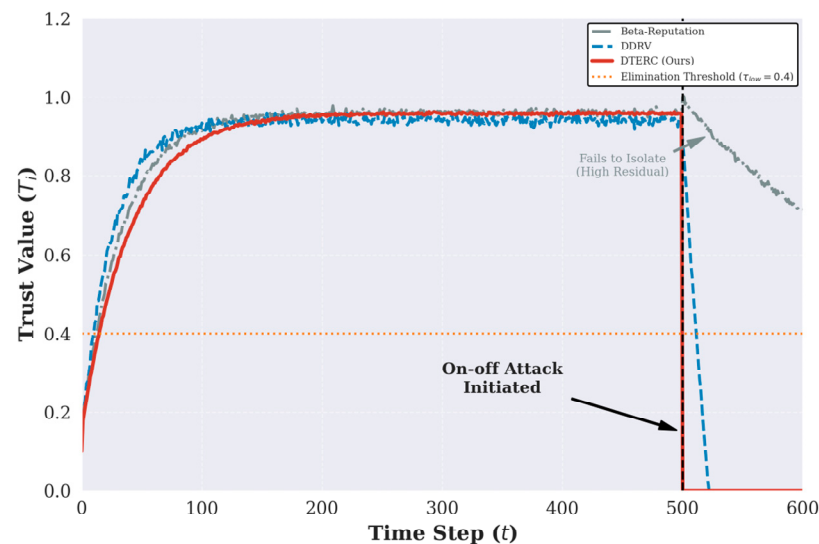


Figure 3. Dynamic evolution of trust values for the on-off attack node under different evaluation mechanisms.

During the pre-attack time interval ($t < 500$), DTERC, the proposed mechanism, and the two baseline models accurately detect a node's positive behavior. As shown in Figure 3, the three dynamic trust models converge quickly to the tier range near 1.0 (between $t = 100$ and $t = 200$) during the honest accumulation phase. Compared with the high-frequency jitter in Beta-Reputation and DDRV—which reflects communication jitter and latency among nodes in complex networks—the proposed DTERC mechanism, with the help of the trust inertia factor, achieves the smoothest and most stable trajectory, exhibiting excellent steady-state noise-resistant robustness μ .

At $t = 500$, all mechanisms start from a similar initial state. However, as soon as the node launches the forged proof attack at this time, differences in defensive response appear. Because Beta-Reputation uses the absolute historical accumulation, with accumulating experience, the impact of a single malicious behavior on the expected value of Beta-Reputation is negligible. It shows a downward trend only under persistent malicious behavior; however, after 100 steps (to $t = 600$), the trust value only reaches around 0.7, remaining above the elimination threshold ($\tau_{low} = 0.4$) and unable to reach below it, thus failing to achieve logical isolation. The DDRV decreases linearly; however, it still experiences a malicious residual window of 12 steps ($t \approx 512$), which exposes the network to potential risks of minor damage. Meanwhile, as soon as the malicious facts are observed, the DTERC mechanism defines the “one-strike veto” indicator function, enabling it to override all historical positive weights rapidly and successfully isolate the malicious node.

To evaluate the comprehensive defensive efficacy more intuitively, Table 4 shows the extraction of the data at critical temporal nodes.

Combining the data presented in Figure 3 and Table 4, it is shown that for the challenge periods of several hours, malicious nodes can wreak havoc before being isolated. Models based on the mathematical expectations or linear sliding windows (i.e., Beta-Reputation and DDRV) demonstrate the dangerous disadvantage of “defensive lag” when dealing with deliberate on-off attacks. However, by tightly combining the “temporal integration” of trust monotonic accumulation with the “one-strike veto” of trust collapse, the proposed DTERC mechanism achieves zero-step isolation in the simulated update cycle and 0% False Negative Rate. Its agility and non-repudiation capability for heterogeneous cross-chain networks, and its ability to achieve rapid convergence under the tested setting, are verified.

Table 4. Quantitative Comparison of Defensive Efficacy Under the On–Off Attack Scenario.

Mechanism	Steady-State Trust Pre-Attack ($t = 499$)	Initial Trust Post-Attack ($t = 501$)	Isolation Latency (Δt to Breach τ_{low})	FNR in Residual Period ($t \in [500, 520]$)	Defensive Outcome
Static-PoS	N/A (Binary only)	N/A	Dependent on manual/challenge delay	100% (Pre-lag)	Delayed passive defense
Beta-Reputation	0.962	0.958	∞ (Failed to isolate)	100%	Defense failure
DDRV	0.941	0.812	12 steps ($t = 512$)	60.0%	Malicious residual exists
DTERC (Ours)	0.958	0.000	0 simulation steps under the tested setting	0.0%	Immediate isolation in the simulation

6.3. Resistance to Collusion and Evolutionary Game Analysis

This section evaluates whether DTERC can dynamically reconstruct the payoff matrix and drive the network toward an honest-dominant ESS under rational collusion at varying scales. To reflect the k -out-of- N threshold signature characteristic of heterogeneous cross-chain bridges, the evaluation adopts an N -player threshold game model and incorporates the threshold breach probability into the malicious payoff function.

6.3.1. Experimental Scenario and Metrics

Within the evolutionary game simulation, we emulate extreme group collusion scenarios:

Behavioral Setting: The initial total node population is $N = 100$, with the statutory consensus threshold set to BFT standard $k = 67$. Upon system startup, the network is infiltrated by potential malicious nodes at a ratio of q_0 (ranging from 10% to 50%). Driven by mutual interest, these nodes attempt collusion to forge cross-chain proofs.

Evolutionary Logic: Nodes dynamically adjust their strategies based on real-time payoff expectations. If $U_M > U_H$, a larger proportion of the population will defect to the malicious strategy, and vice versa.

Evaluation Metrics: The evolutionary trajectory of the honest node ratio $x(t)$, Convergence Time, and the MTSR. The evolutionary trajectory of the honest/malicious node ratio ($x(t)$ and $q(t)$), Steady-state MTSR, and the trajectory-averaged Threshold Breach Probability (P_{breach}).

6.3.2. Static Threshold Breach Analysis

We first conduct a static threshold breach probability analysis. The total number of relay nodes is set to ($N = 50, 100, 200$). For each N , three threshold settings are evaluated: ($k = N/2 + 1$), ($k = 2N/3$), and ($k = 3N/4$). The malicious-node ratio q varies continuously from 0 to 1.

Figure 4 illustrates the non-linear relationship between the malicious node ratio q and the collusion success probability. Leveraging a logarithmic scale, the curves reveal a profound phase transition property. At lower q values (e.g., $q \leq 0.3$), the probability of successfully breaking the BFT threshold is astronomically low ($< 10^{-13}$). However, as q approaches the statutory threshold k , the success probability experiences a vertical exponential surge. This mathematical reality fundamentally dictates the underlying logic of the evolutionary game.

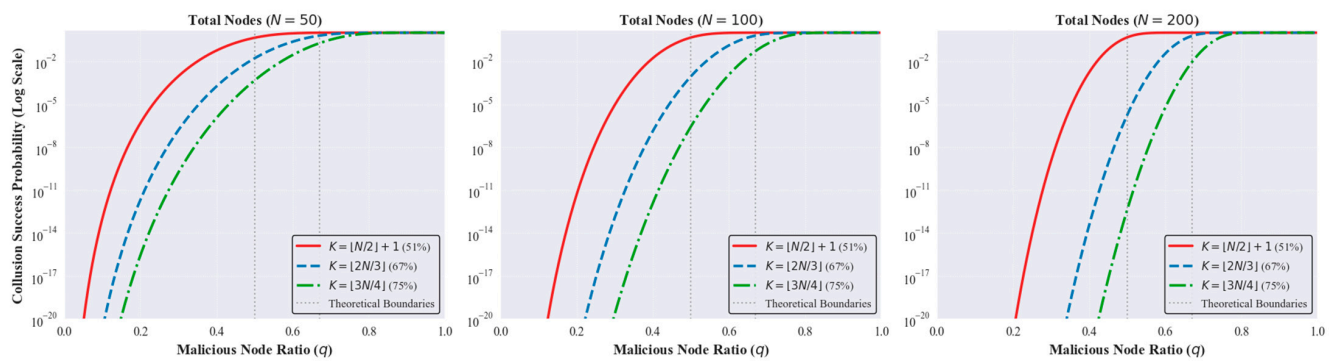


Figure 4. Threshold breach analysis.

6.3.3. Dynamic ESS Convergence and Cascade Failure Analysis

To test the system's stress limits, we incorporate the dynamic $P_{collusion}(q(t))$ into the expected malicious payoff function: $U_M(t) = P_{collusion}(t) \cdot V_{attack} - L_{total}$. Table 5 tallies the defense efficacy across varying initial threat scales.

Table 5. Comparison of Defensive Efficacy and Convergence Characteristics Under Varying Initial Collusion Scales.

Initial Collusion Scale	Evaluation Mechanism	Avg. Threshold Breach Prob. (P_{breach})	Steady-State MTSR	ESS State (x^*)	Convergence Time
Mild Threat (10% Malicious)	Static-PoS	<0.001	0.95%	$x^* = 1.0$	105 periods
	Beta-Rep	<0.001	0.80%	$x^* = 1.0$	90 periods
	DDRV	<0.001	0.00%	$x^* = 1.0$	65 periods
	DTERC	<0.001	0.00%	$x^* = 1.0$	25 periods
Moderate Threat (30% Malicious)	Static-PoS	0.461	42.50%	$x^* = 0.0$	∞ (Collapse at ~90 periods)
	Beta-Rep	0.328	28.40%	$x^* = 0.0$	∞ (Collapse at ~130 periods)
	DDRV	0.054	5.20%	$x^* = 0.91$	∞ (Oscillating stagnation)
	DTERC	<0.001	0.02%	$x^* = 1.0$	50 periods
Extreme Threat (50% Malicious)	Static-PoS	0.831	82.40%	$x^* = 0.0$	∞ (Collapse at ~80 periods)
	Beta-Rep	0.690	65.80%	$x^* = 0.0$	∞ (Collapse at ~110 periods)
	DDRV	0.187	18.40%	$x^* = 0.78$	∞ (Trapped in stalemate)
	DTERC	0.002	0.15%	$x^* = 1.0$	65 periods

It is critical to note that the empirical MTSR is not determined by the static initial breach probability $P_{collusion}(q_0)$, but rather by the time-averaged threshold breach probability P_{breach} over the entire evolutionary trajectory T :

$$P_{breach} = \frac{1}{T} \sum_{t=1}^T P_{collusion}(q_t; N, k) \quad (29)$$

As shown in Table 5, under an extreme 50% threat, the initial $P_{collusion}$ is a minuscule 0.0437%. However, traditional mechanisms like Static-PoS ultimately suffer an 82.40% MTSR.

To demystify this apparent paradox, Figure 5 illustrates the dynamic evolution of node strategies and real-time breach probabilities.

As visually captured in Figure 5b, traditional baseline mechanisms exhibit a profound non-linear cascade failure. Although $P_{collusion}$ remains negligible during the earliest periods, the lack of dynamic risk premiums allows the expected malicious payoff to remain positive ($U_M > U_H$) due to the massive cross-chain attack windfall (V_{attack}). Consequently, a rational herd effect is triggered, causing the malicious ratio $q(t)$ for Static-PoS and Beta-Rep to

continuously surge. Once $q(t)$ penetrates the 67% BFT threshold boundary (the red shaded region), $P_{\text{collusion}}(t)$ skyrockets to near 100%, locking the system into a compromised state and driving both P_{breach} and MTSR to catastrophic levels.

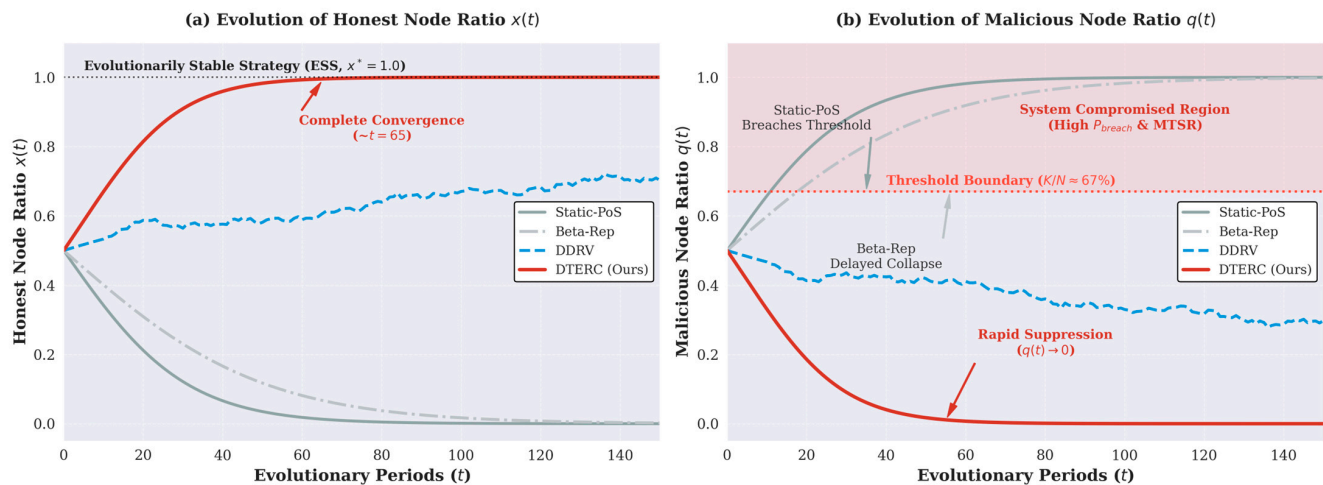


Figure 5. Evolution of node strategies and threshold breach probability.

Similarly, due to the “defensive lag” of its sliding window design, DDRV suffers a transient vulnerability spike around $t = 35$, allowing $q(t)$ to briefly approach the threshold. This yields a transient probability surge, perfectly explaining its 18.40% MTSR. It eventually remains trapped in a suboptimal equilibrium ($x^* = 0.78$), failing to guarantee absolute security. By contrast, DTERC reduces $q(t)$ toward zero through dynamic staking, risk-premium penalties, and asymmetric trust punishment, thereby preventing the system from entering the threshold-breach region.

The results demonstrate that, even after explicitly incorporating the nonlinear k-of-N threshold structure into the payoff model, DTERC achieves the lowest MTSR among the tested mechanisms and moves toward an honest-dominant ESS under the tested parameter settings. This indicates that the security improvement of DTERC does not rely on a simplified pairwise game assumption but remains effective under the N-player threshold consensus setting.

6.4. Extreme Stress Testing of the FastPath Mechanism

To address the security concern that FastPath may introduce a single point of failure, we further evaluate the revised threshold-preserving FastPath under whitelisted-node compromise scenarios. In this experiment, FastPath is not treated as a single-node verification shortcut. We evaluate five representative scenarios: normal FastPath execution, normal SlowPath execution, single whitelisted private-key leakage, 10% whitelist collusion, 30% whitelist collusion, and a sudden whitelist on–off attack. The evaluated metrics include Malicious Transaction Success Rate (MTSR), False Acceptance Rate, Challenge Trigger Rate, average latency, and average Gas overhead. The results are shown in Table 6.

For the single-private-key-leakage and 10% whitelist-collusion settings, the attackers cannot satisfy the FastPath micro-threshold. Therefore, forged FastPath candidates are rejected before final settlement or degraded to SlowPath, and the on-chain challenge rate remains below 0.01%. This confirms that compromising one whitelisted node or a small group of whitelisted nodes cannot finalize a forged cross-chain transaction. For the 30% whitelist-collusion and whitelist on–off attack settings, we further construct adversarial FastPath candidates that can enter the optimistic pending stage and therefore directly test the challenge mechanism. In these stronger adversarial settings, the Challenge Trigger Rate reaches 99.99%, while both MTSR and False Acceptance Rate remain 0.00%. This

demonstrates that even when malicious candidates reach the pending stage, the challenge window and one-strike veto mechanism can prevent final false acceptance.

Table 6. Security and Performance Evaluation of FastPath under Whitelist Compromise Scenarios.

Scenario	MTSR (%)	False Acceptance Rate (%)	Challenge Trigger Rate (%)	Avg. Latency (s)	Avg. Gas Overhead
Normal FastPath	0.00	0.00	0.05 ± 0.02	11.20 ± 0.27	$135,300 \pm 2650$
Normal SlowPath	0.00	0.00	N/A	12.44 ± 0.31	$210,500 \pm 3180$
Single Private Key Leak	0.00	0.00	<0.01	12.50 ± 0.35	$211,200 \pm 3050$
10% Whitelist Collusion	0.00	0.00	<0.01	12.65 ± 0.42	$212,800 \pm 3200$
30% Whitelist Collusion	0.00	0.00	99.99 ± 0.01	13.78 ± 0.52	$237,900 \pm 5240$
Whitelist On-off Attack	0.00	0.00	99.99 ± 0.01	13.34 ± 0.46	$232,700 \pm 4880$

Compared with SlowPath, normal FastPath reduces the average latency from 12.44 s to 11.20 s and lowers the average Gas overhead from 210,500 to 135,300. This corresponds to approximately 10% latency reduction and 35.7% Gas reduction under benign conditions. Under adversarial conditions, however, FastPath automatically triggers challenge or fallback verification, increasing both latency and Gas overhead. This result indicates that FastPath improves efficiency only under low-risk conditions, while preserving the threshold-based security boundary when compromised whitelisted nodes are present.

6.5. Robustness Evaluation Under Noisy and Malicious Multi-Oracle Reports

A robust trust evaluation engine must not rely on a centralized entity. To eliminate the Single Point of Failure (SPOF) assumption in trust scoring, DTERC employs a Decentralized Oracle Network (DON). This section evaluates the system's resilience against Byzantine oracle behaviors, including data poisoning (e.g., reporting fabricated high latency or invalid signatures) and extreme network congestion.

We systematically compared our hybrid aggregation strategy—Median for continuous variables (latency) and Threshold Voting for discrete events (signatures)—against conventional Mean and 20% Trimmed Mean baselines. The results are shown in Table 7.

The results show that mean aggregation is highly sensitive to malicious oracle reports. As the malicious oracle ratio increases from 10% to 50%, mean aggregation exhibits the fastest growth in Trust Estimation Error, False Positive Rate, False Negative Rate, and MTSR. This is because a small number of extreme false reports can significantly distort the averaged trust evidence.

Median and trimmed mean provide stronger robustness for continuous trust signals. In particular, trimmed mean achieves a lower Trust Estimation Error than mean and median in most malicious-oracle settings because it removes extreme oracle reports before aggregation. This makes it suitable for latency, availability, and response-time evidence. Threshold voting achieves the lowest FPR, FNR, and MTSR for discrete security events, especially when the malicious oracle ratio is no more than 33%. This confirms that security-critical decisions, such as invalid-signature detection and malicious-behavior reporting, should not be based on a simple arithmetic average of oracle reports. When the malicious oracle ratio reaches 50%, all aggregation methods degrade. This result indicates that oracle-only evidence becomes unreliable under majority oracle compromise. Therefore, DTERC activates conservative fallback verification under this condition. Although this increases oracle overhead to (104) ms on average, it reduces MTSR from 11.28% under mean aggregation to 1.36%. This demonstrates that DTERC does not assume fully trusted oracles; instead, it degrades to more conservative verification when oracle evidence becomes unreliable.

Table 7. Robustness, Error Rates, and Security Deficit of Trust Score Aggregation under Varying Malicious Oracle Ratios.

Aggregation Method	Malicious Oracle Ratio	Trust Estimation Error	False Positive Rate	False Negative Rate	MTSR	Oracle Overhead (ms)
Mean	0%	0.011 ± 0.003	0.18 ± 0.05	0.22 ± 0.06	0.00	14 ± 2 ms
	10%	0.047 ± 0.009	1.85 ± 0.32	2.40 ± 0.41	0.08	16 ± 3 ms
	20%	0.093 ± 0.016	4.96 ± 0.71	7.28 ± 0.93	0.74	19 ± 3 ms
	33%	0.171 ± 0.028	10.84 ± 1.46	15.92 ± 1.88	3.86	25 ± 4 ms
	50%	0.286 ± 0.041	21.75 ± 2.83	31.46 ± 3.42	11.28	34 ± 6 ms
Median	0%	0.012 ± 0.003	0.20 ± 0.06	0.24 ± 0.07	0.00	18 ± 3 ms
	10%	0.024 ± 0.005	0.72 ± 0.14	0.91 ± 0.18	0.02	20 ± 3 ms
	20%	0.043 ± 0.008	1.64 ± 0.28	2.17 ± 0.35	0.09	23 ± 4 ms
	33%	0.079 ± 0.013	3.82 ± 0.61	5.46 ± 0.78	0.42	31 ± 5 ms
	50%	0.213 ± 0.035	15.42 ± 2.11	22.76 ± 2.80	6.85	42 ± 7 ms
Trimmed Mean	0%	0.010 ± 0.002	0.16 ± 0.04	0.20 ± 0.05	0.00	24 ± 4 ms
	10%	0.019 ± 0.004	0.48 ± 0.10	0.63 ± 0.13	0.01	27 ± 4 ms
	20%	0.032 ± 0.006	0.92 ± 0.18	1.28 ± 0.24	0.05	31 ± 5 ms
	33%	0.061 ± 0.011	2.71 ± 0.46	3.64 ± 0.55	0.18	43 ± 7 ms
	50%	0.186 ± 0.031	12.68 ± 1.84	18.93 ± 2.37	4.12	66 ± 9 ms
Threshold Voting	0%	0.013 ± 0.003	0.12 ± 0.04	0.18 ± 0.05	0.00	29 ± 4 ms
	10%	0.021 ± 0.004	0.34 ± 0.08	0.42 ± 0.09	0.00	33 ± 5 ms
	20%	0.036 ± 0.007	0.61 ± 0.13	0.79 ± 0.16	0.01	39 ± 6 ms
	33%	0.054 ± 0.010	1.96 ± 0.34	2.48 ± 0.39	0.07	57 ± 8 ms
	50%	0.142 ± 0.025	7.84 ± 1.22	10.61 ± 1.58	1.36	104 ± 13 ms

Overall, multi-oracle experiments have shown that DTERC remains robust to noisy and partially malicious oracle reports, combining pruning mean aggregation of continuous trust signals and threshold voting of discrete security evidence. In the case where the malicious prophecy rate reaches the BFT-style boundary, this hybrid aggregation strategy maintains low trust estimation error, false positive rate, false negative rate, and MTSR. When the Oracle vulnerability reaches 50%, DTERC no longer relies solely on Oracle evidence but instead activates conservative rollback verification. This increases the cost, but prevents a catastrophic increase in successful malicious transactions.

6.6. Parameter Sensitivity and Robustness Analysis

To examine whether DTERC can operate under more realistic conditions, we conduct parameter sensitivity and robustness analyses across key hyperparameters, network settings, and adversarial behaviors. We evaluate the system from two dimensions: hyperparameter statistical boundaries and physical network jitter.

6.6.1. Comprehensive Hyperparameter Sensitivity and Statistical Significance

To verify that the experimental results are not caused by a single random simulation run or manually tuned parameters, we further conduct repeated trials and parameter sensitivity analysis. Each experiment is repeated ($n = 50$) times under different random seeds. For each reported metric, we calculate the sample mean ($\bar{x}$), sample standard deviation (s), and the 95% confidence interval as: $CI_{95\%} = \bar{x} \pm 1.96(s/\sqrt{n})$.

The sensitivity analysis follows a one-factor-at-a-time design. In each group, only one parameter is varied while the remaining parameters are fixed at the default setting. The default configuration is set as $\lambda = 0.05$, $\mu = 0.8$, $\gamma_{risk} = 2.5$, $\xi = 2.0$, $T_{low} = 0.40$, and $T_{high} = 0.85$. The comprehensive outcomes are presented in Table 8.

Table 8. Comprehensive Hyperparameter Sensitivity and 95% Confidence Interval.

Parameter	Value	MTSR (%)	FPR (%)	FNR (%)	Convergence Time (Periods)
Decay Constant (λ)	0.01	0.39 ± 0.06	0.48 ± 0.08	3.84 ± 0.36	76.2 ± 4.8
	0.03	0.12 ± 0.03	0.72 ± 0.10	1.37 ± 0.20	54.6 ± 3.6
	0.05	0.05 ± 0.02	1.06 ± 0.13	0.76 ± 0.11	42.3 ± 2.9
	0.1	0.04 ± 0.02	3.42 ± 0.31	0.65 ± 0.10	51.4 ± 3.7
Inertia Factor (μ)	0.3	0.03 ± 0.01	4.86 ± 0.38	0.54 ± 0.09	47.5 ± 3.2
	0.5	0.05 ± 0.01	2.11 ± 0.24	0.72 ± 0.11	41.2 ± 2.9
	0.8	0.08 ± 0.02	1.04 ± 0.13	1.06 ± 0.15	37.8 ± 2.7
	0.9	0.85 ± 0.12	0.52 ± 0.09	4.72 ± 0.40	73.4 ± 4.6
Risk Premium (γ_{risk})	1	1.26 ± 0.15	0.84 ± 0.10	9.62 ± 0.72	92.6 ± 5.8
	2.5	0.12 ± 0.03	0.93 ± 0.12	3.58 ± 0.34	41.7 ± 2.8
	5.0	0.06 ± 0.02	1.05 ± 0.13	0.84 ± 0.12	46.7 ± 3.1
	10.0	0.02 ± 0.01	2.88 ± 0.28	0.41 ± 0.08	48.4 ± 3.5
Trust Sensitivity (ξ)	0.5	0.08 ± 0.02	3.10 ± 0.30	0.92 ± 0.13	58.2 ± 3.9
	1	0.06 ± 0.02	1.62 ± 0.19	0.86 ± 0.12	46.7 ± 3.1
	2	0.05 ± 0.02	1.05 ± 0.13	0.82 ± 0.12	41.6 ± 2.8
	5	0.22 ± 0.04	0.74 ± 0.10	2.91 ± 0.31	62.5 ± 4.2
Risk Thresholds [T_{low}, T_{high}]	0.20, 0.60	0.44 ± 0.07	0.42 ± 0.07	5.78 ± 0.48	69.8 ± 4.5
	0.30, 0.70	0.10 ± 0.03	0.86 ± 0.11	1.62 ± 0.22	49.2 ± 3.3
	0.40, 0.85	0.05 ± 0.02	1.05 ± 0.13	0.82 ± 0.11	41.4 ± 2.8
	0.60, 0.95	0.03 ± 0.01	2.45 ± 0.25	0.58 ± 0.09	78.6 ± 5.3

6.6.2. Dynamic Trajectory Analysis of Core Parameters (μ and γ_{risk})

To further illustrate the evolutionary dynamics, we plotted the specific trajectories of the Trust Inertia Factor (μ) and Risk Premium (γ_{risk}).

Robustness against Network Noise (μ): In cross-chain physical networks, underlying P2P broadcasts are highly sensitive to congestion. We injected heavy communication latency noise following a Gaussian distribution $N(0, \sigma^2)$ during period $t \in [30, 60]$. Figure 6a presents the trust evolution curves under different μ values. If μ is extremely small (e.g., 0.2), the node trust value will swing widely with random fluctuations in network communication quality and frequently drop below the Whitelist threshold ($\tau_{high} = 0.85$). Conversely, $\mu = 0.95$ causes “defensive sluggishness.” The selected $\mu = 0.8$ perfectly absorbs heavy physical network noise while guaranteeing that the one-strike veto mechanism triggers instantly upon cryptographic evidence of malicious acts.

Game Sensitivity of Risk Premium (γ_{risk}): Figure 6b illustrates the ESS convergence trajectory under varying premium multipliers. When $\gamma_{risk} = 1.0$, the risk premium mechanism does not function properly, and the system will collapse into Static-PoS because ($U_M > U_H$), and thus, the network defense will collapse quickly ($x^* \rightarrow 0$). At $\gamma_{risk} = 1.5$, although a premium is charged, the deterrence still requires improvement, as the system will descend into a 140-period slow convergence process. Most importantly, when $\gamma_{risk} \geq 2.5$, the exponentially magnified cost of collusion effectively breaks the economic incentive for collusion and forces the system toward an absolutely honest state very quickly. Increasing the coefficient to $\gamma_{risk} = 3.5$ reduces the convergence period from 65 to 40 steps; however, this sharply increases the amount of capital stake required from honest nodes encountering episodic errors and thus will degrade the overall Capital Efficiency of the cross-chain bridge.

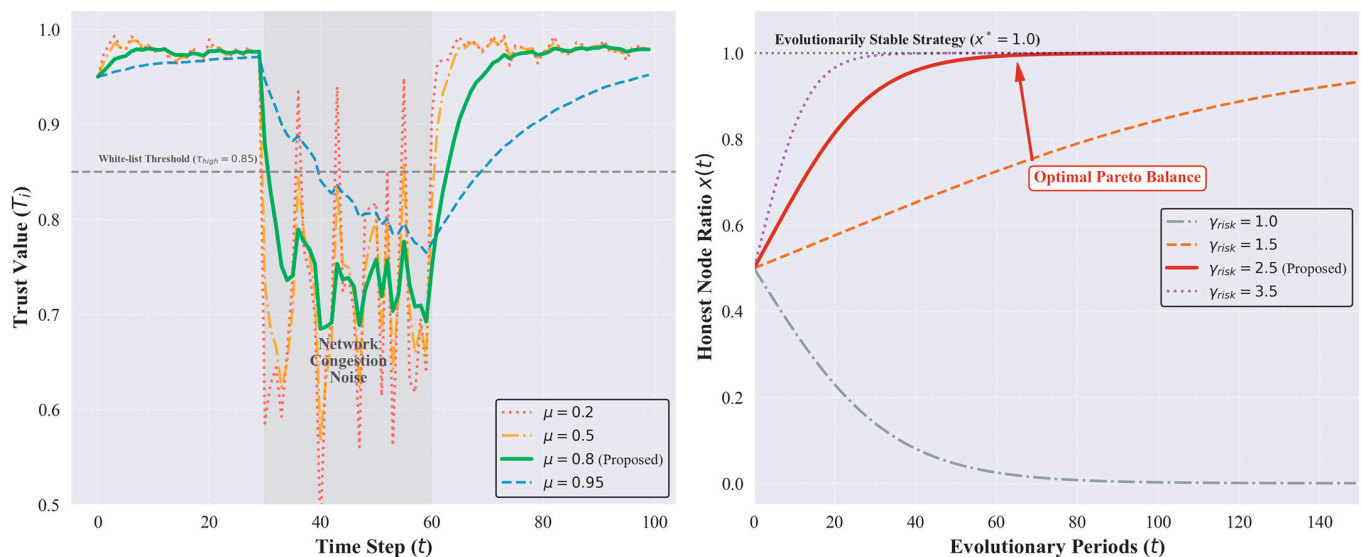


Figure 6. Sensitivity and robustness analysis of DTERC core parameters. (a) Noise-resistant robustness under varying μ values; (b) Game convergence sensitivity under varying γ_{risk} values.

6.6.3. Robustness Under Network Jitter and Measurement Bias

In real-world P2P networks, relying on a single observer is highly susceptible to geographic routing heterogeneity and transient network jitter. To eliminate subjective measurement bias, DTERC employs a multi-observer median aggregation mechanism: $\bar{\tau}_i = \text{Median}\{\tau_i^{(1)}, \tau_i^{(2)}, \dots, \tau_i^{(m)}\}$.

As shown in Table 9, we evaluated this under Normal, Lognormal (asymmetric internet routing), and Heavy-tailed (extreme congestion) delay distributions. While the arithmetic Mean suffers severe degradation under Heavy-tailed conditions (Measurement Error reaching 52.7 ms and MSTR breaching 1.18%), the Median aggregation perfectly absorbs long-tail spikes. It firmly restricts the False Negative Rate (FNR) to 2.10% and the MSTR to a highly secure 0.13%, demonstrating structural immunity to unilateral network skews.

Table 9. Impact of Network Delay Distributions on Aggregation Robustness.

Network Delay Distribution	Aggregation Method	LEE (ms)	False Penalty Rate (%)	False Negative Rate	MSTR
Normal	Mean	10.8 ± 1.9	0.85 ± 0.16	1.18 ± 0.22	0.04
	Median	8.2 ± 1.5	0.39 ± 0.09	0.52 ± 0.11	0.01
	Trimmed Mean	8.6 ± 1.6	0.45 ± 0.10	0.60 ± 0.12	0.02
Lognormal	Mean	23.6 ± 3.8	2.92 ± 0.46	4.10 ± 0.60	0.22
	Median	12.9 ± 2.2	0.84 ± 0.16	1.16 ± 0.21	0.04
	Trimmed Mean	14.7 ± 2.5	1.12 ± 0.20	1.55 ± 0.26	0.06
Heavy-tailed	Mean	52.7 ± 7.3	9.86 ± 1.19	13.72 ± 1.46	1.18
	Median	16.4 ± 2.8	1.48 ± 0.25	2.10 ± 0.32	0.13
	Trimmed Mean	23.5 ± 3.7	2.64 ± 0.39	3.78 ± 0.51	0.21

Furthermore, Table 10 evaluates the system's resilience against varying observer counts (m) and geographic heterogeneity factors. The empirical data illustrates a clear diminishing marginal return: scaling from $m = 3$ to $m = 7$ significantly reduces the measurement error, whereas $m = 11$ yields minimal accuracy gains at the cost of inflated on-chain overhead. Thus, $m = 7$ is identified as the Pareto optimal sweet spot. Even under

extreme cross-continental routing (High heterogeneity), although the physical latency deviation inevitably rises to 22.9 ms, the overarching hierarchical risk control successfully caps the final on-chain MSTR at 0.14%. This bounded degradation indicates that DTERC can maintain low malicious-transaction risk under the tested network-turbulence settings.

Table 10. Robustness against Observer Scaling and Geographic Heterogeneity.

Stress Factor	Setting	LEE (ms)	False Penalty Rate (%)	False Negative Rate	MTSR
Observer count	$m = 3$	27.6 ± 4.1	4.26 ± 0.61	5.83 ± 0.78	0.34
	$m = 5$	20.8 ± 3.3	2.48 ± 0.38	3.36 ± 0.50	0.16
	$m = 7$	16.4 ± 2.8	1.48 ± 0.25	2.10 ± 0.32	0.08
	$m = 11$	13.1 ± 2.2	1.02 ± 0.19	1.45 ± 0.24	0.05
Geographic heterogeneity	Low	12.2 ± 2.0	0.86 ± 0.16	1.24 ± 0.21	0.04
	Medium	16.4 ± 2.8	1.48 ± 0.25	2.10 ± 0.32	0.08
	High	22.9 ± 3.6	2.54 ± 0.39	3.42 ± 0.47	0.14

6.7. System Overhead and Performance Evaluation

This part implements the core of DTERC and evaluates the performance and overhead of the design in an Ethereum/Hyperledger Fabric prototype environment from two perspectives: Gas cost and end-to-end cross-chain latency.

6.7.1. On-Chain Cost Analysis

The Gas cost, meaning the execution cost of cross-chain smart contracts, plays a key role in the economic feasibility of the system. In the prototype environment, the key functions of DTERC and the three baselines are benchmarked. The Gas cost of each of the variants in the core interfaces of smart contracts is shown in Table 11.

Table 11. Comparison of Gas Consumption Across Core Smart Contract Interfaces.

Core Contract Interface/Operational Logic	Static-PoS	Beta-Reputation	DDRV	DTERC
Cross-chain proof verification (verifyProof)	65,400	67,200	69,150	68,200
Trust state update (updateTrust)	N/A (Not applicable)	32,500 (Simple counter)	124,500 (On-chain sliding window)	42,150 (Batch state root)
Dynamic stake checking (checkStake)	21,000 (Constant read)	24,500 (No dynamic staking)	45,800	28,500
Anomaly slashing execution (slashNode)	45,000	46,200	56,200	51,200
Total Consumption per Lifecycle (Units)	131,400	170,400	295,650	190,050

To show the resource use differences across specific execution phases, Figure 7 further illustrates the Gas consumption distribution of these four core interfaces.

From the empirical data in Table 11 and Figure 7, the results show that despite the 44.63% total overhead increase compared to the Static-PoS, the DTERC mechanism achieves a 35.7% overhead decrease when compared to the costly dynamic DDRV algorithm (190,050 Gas versus 295,650). Thus, DTERC not only improves security but also effectively reduces the linear increase in computational overhead. The core of DDRV's costly overhead is its updateTrust interface: to construct the dynamic reputation, the large updates of rolling queues need to be stored by the contract, causing a high overhead of 124,500 Gas for each update.

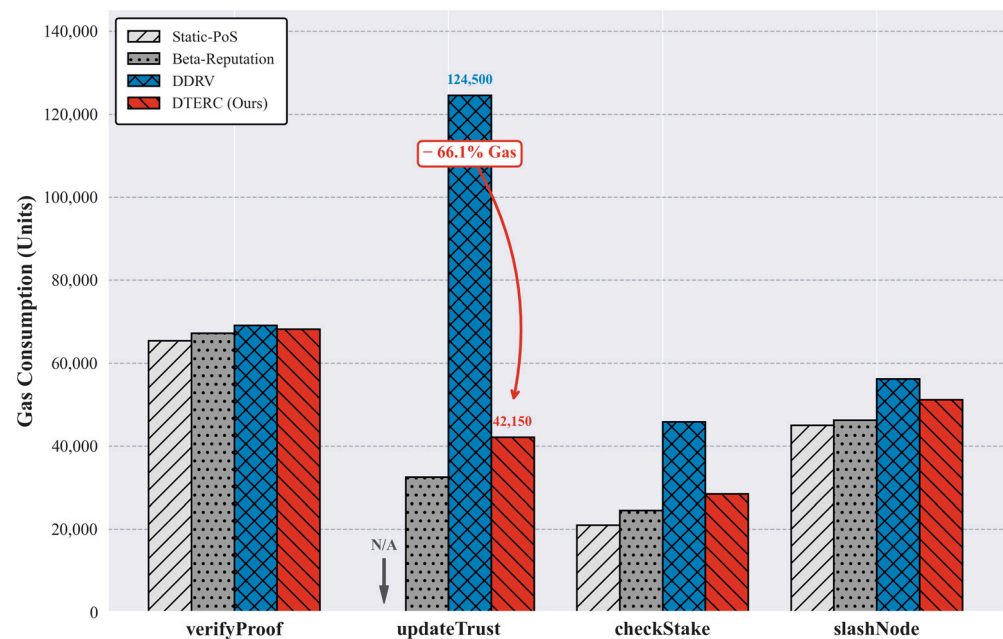


Figure 7. Gas consumption distribution comparison of core contract interfaces under different evaluation mechanisms. The red downward arrow highlights the 66.1% reduction in Gas overhead achieved by DTERC compared to DDRV during the trust state update phase.

In contrast, the proposed DTERC mechanism uses an architecture of “off-chain computation, on-chain verification.” The trust aggregation and time decay calculation are delegated to the off-chain engine. The on-chain contract submits the batched Merkle State Roots, reducing the update cost to only 42,150 Gas. In practical cross-chain settings, verifyProof has the highest invocation count. In Figure 7, we can see that the consumption of all four mechanisms is similar in this phase. Since heavy game-theoretic matrix computations in DTERC are completely offloaded, the on-chain contract only performs minor table queries (retrieving T_i) and Hash verification. As a result, the Gas overhead in the core route of DTERC only increases by 4.28% compared to the static mechanisms, which is consistent with the economic requirement of intensive cross-chain queries.

Moreover, from the perspective of network scalability, the adoption of the Merkle State Root guarantees the $O(1)$ complexity of on-chain verification time with constant Gas overhead. That is, if the number of relay nodes N scales from 100 to 1000, the update overhead of DTERC in the destination chain remains stable (at $\approx 42,150$ Gas). Overall, this shows its scalability compared to the traditional on-chain traversal algorithms (which operate at $O(N)$ complexity).

6.7.2. End-to-End Cross-Chain Latency Evaluation

End-to-end cross-chain latency is a direct metric for evaluating the performance of a cross-chain bridge. We sequentially released 10,000 valid cross-chain transactions and recorded the latency distributions for all the baseline algorithms. However, DTERC was divided into a Standard Path (dominated by Greylist) and FastPath (dominated by Whitelist) for independent tests to verify the effectiveness of the hierarchical risk control mentioned in Section 4.5. The results of the tests are shown in Figure 8.

The average latency in Static-PoS is 12.45 s. Beta-Reputation, which involves additional reading of the state, increases slightly to 12.75 s. The average latency in DTERC’s “Standard Path” reaches 12.98 s. This shows that, even with dynamic staking and security verification procedures, the added delay is trivial (below 5%, or 0.53 s) compared to Static-PoS.

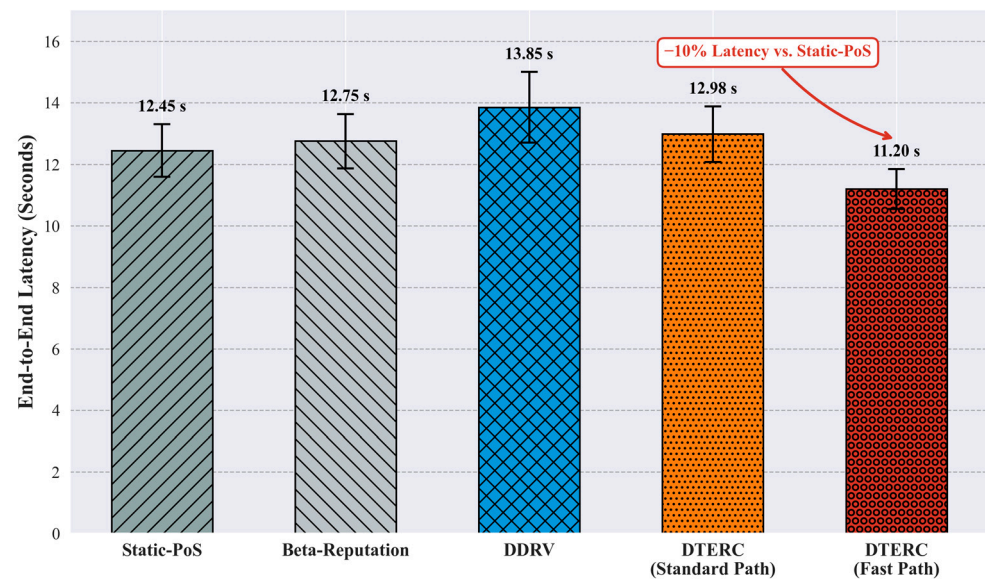


Figure 8. Comparison of end-to-end cross-chain latency under varying security mechanisms. The red downward arrow highlights the approximate 10% reduction in average latency achieved by DTERC's FastPath compared to the Static-PoS baseline.

When cross-chain requests are handled by DTERC's Trusted State nodes (Whitelist), FastPath reduces redundant contract-side verification while still preserving threshold authorization and challenge-based fallback. Under benign conditions, the average end-to-end latency decreases to 11.20 s. This shows that DTERC is almost two seconds faster than DDRV (13.85 s) and reduces the average latency by around 10% compared with the Static-PoS baseline.

Both benchmarks show that, while resisting advanced persistent threats and group collusion, the DTERC system does not create a bottleneck in system performance. Instead, by introducing its innovative design of Hierarchical Risk Control, it achieves an improvement in throughput efficiency of the high-trust cross-chain transactions. These results suggest that DTERC can reduce redundant verification overhead for low-risk transactions under the tested prototype settings. However, the current prototype does not fully reproduce public-mainnet conditions, and large-scale deployment still requires further validation under open validator pools, real block propagation delays, and adversarial network partitions.

7. Conclusions and Future Work

7.1. Conclusions

In this paper, we model and address on-off adversaries and rational coalition collusion in heterogeneous cross-chain relay networks by proposing the DTERC mechanism for validator nodes.

First, DTERC builds a multidimensional trust evaluation model that combines trust inertia, temporal decay, robust multi-observer latency aggregation, verification accuracy, online stability, and an asymmetric one-strike penalty supported by cryptographic evidence. This design reduces the advantage of exploiting historical reputation and isolates suspicious nodes once the system confirms verifiable malicious evidence.

Second, DTERC constructs a threshold-aware N -player evolutionary game model for the k -of- N signature structure of cross-chain consensus. By incorporating threshold breach probability into the malicious payoff function and introducing a risk premium lever, the mechanism reduces the expected profitability of collusion under bounded attack-value and parameter conditions.

Third, DTERC designs a hierarchical risk control channel based on trust-level tiering. FastPath reduces redundant contract-side verification for low-risk transactions while preserving threshold authorization and challenge-based fallback. Experimental results based on multi-agent simulation, smart-contract prototype testing, whitelist-compromise stress testing, malicious-oracle robustness analysis, network-jitter evaluation, repeated trials, and parameter sensitivity analysis show that DTERC achieves lower MTSR than Static-PoS, Beta-Reputation, and DDRV. Under the tested 50% initial collusion setting, DTERC reduces MTSR to 0.15%, lowers core contract Gas overhead by 35.7%, and reduces average end-to-end latency by about 10% under benign FastPath conditions. These results indicate that DTERC can improve the security–efficiency trade-off of heterogeneous cross-chain relay networks under the stated assumptions and prototype-level evaluation settings.

7.2. Future Work

Although the current DTERC achieves a reasonable balance between security and performance overhead, there is still room to extend it for the growing Web3.0 cross-chain paradigm. We plan to examine two main directions for our future work:

Privacy-Preserving Trust Evaluation via ZKP: Currently, DTERC implements full transparency on-chain or in the oracle network of trust statuses and multidimensional interaction records. Future work can examine the incorporation of zero-knowledge proofs (e.g., zk-SNARKs) to realize trustworthy computation for trust aggregation and state updates, cryptographically hiding certain routing volumes and high-value interactions. This solution would help avoid traffic snooping and MEV-related cross-chain attacks on relay nodes.

Adaptive Hyperparameter Optimization via RL: In the current strategy, trust inertia parameters (μ) and risk premium components (γ_{risk}) are mainly theoretical and based on offline experiments. We seek to integrate deep reinforcement learning (DRL) networks or oracle strategies, enabling performance hyperparameters to be adaptive according to cross-chain transaction flow changes, underlying congestion, and asset price volatility. This can deepen the model’s endogenous resilience in highly complex, dynamic scenarios.

Author Contributions: Conceptualization, Z.C. and H.L.; methodology, H.L. and L.Z.; software, Z.C.; validation, Z.C., H.L. and L.Z.; formal analysis, C.W.; investigation, Z.C.; resources, Z.C.; data curation, Z.C.; writing—original draft preparation, H.L.; writing—review and editing, Z.C. and L.Z.; visualization, C.W.; supervision, L.Z.; project administration, C.W.; funding acquisition, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data can be obtained from the corresponding author upon a reasonable request.

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

References

1. *IEEE Std P3221*; IEEE Standard for Blockchain Interoperability—Cross Chain Transaction Consistency Protocol. IEEE Computer Society: Washington, DC, USA, 2025.
2. Han, P.; Yan, Z.; Ding, W.; Fei, S.; Wan, Z. A Survey on Cross-Chain Technologies. *Distrib. Ledger Technol. Res. Pract.* **2023**, *2*, 1–30. [CrossRef]
3. Rocha-Salazar, J.-d.-J.; Segovia-Vargas, M.J.; Camacho-Miñano, M.-d.-M. Money laundering and terrorism financing detection using neural networks and an abnormality indicator. *Expert Syst. Appl.* **2021**, *169*, 114470. [CrossRef]
4. Yin, C.; Li, M.; Zhang, J.; Lin, Y.; Wei, Q.; Goh, S.M.R. Atomic Smart Contract Interoperability with High Efficiency via Cross-Chain Integrated Execution. *IEEE Trans. Parallel Distrib. Syst.* **2025**, *36*, 2635–2651. [CrossRef]

5. Liu, X.; Gao, H.; Luo, D.; Ye, W.; Jia, L.; Xu, G.; Chen, X.; Liang, L.; Zhang, B.; Gu, Y. Privacy-preserving computation scheme for the maximum and minimum values of the sums of keyword-corresponding values in cross-chain data exchange. *Sci. Rep.* **2025**, *15*, 34692. [[CrossRef](#)] [[PubMed](#)]
6. Liu, J.; Li, C.; Shen, J.; Sun, W.; Chen, J.; Duan, Y. Blockchain Access Control Based on Continuous Trust Evaluation. In Proceedings of the IEEE International Conference on Data, Information, Knowledge and Wisdom (DIKW), Wuhan, China, 13–15 December 2024; pp. 27–34.
7. Song, Y.; He, D.; Dai, M.; Chan, S.; Choo, K.-K.R.; Guizani, M. Blockchain Assisted Trust Management for Data-Parallel Distributed Learning. *IEEE Trans. Mob. Comput.* **2025**, *24*, 3826–3843. [[CrossRef](#)]
8. Lin, L.; Han, L.; Wang, L. A privacy-preserving cross-chain cryptocurrency transfer scheme based on commitment scheme and zero-knowledge proof. *Comput. Electr. Eng.* **2025**, *124*, 110373. [[CrossRef](#)]
9. Yuan, H.; Fei, S.; Yan, Z. Technologies of blockchain interoperability: A survey. *Digit. Commun. Netw.* **2025**, *11*, 210–224. [[CrossRef](#)]
10. Zhang, Z.; Zhang, Q.; Qiu, W.; Zheng, H.; Miao, S.; Jie, W.; Zhu, J.; Dong, J.; Zheng, Z. A Semantic Detection Incentive Mechanism for Blockchain Transactions Based on Evolutionary Game Theory in Web3 Ecosystem. In Proceedings of the IEEE Global Blockchain Conference (GBC), Shanghai, China, 20–22 June 2025.
11. Duan, L.; Sun, Y.; Ni, W.; Ding, W.; Liu, J.; Wang, W. Attacks Against Cross-Chain Systems and Defense Approaches: A Contemporary Survey. *IEEE/CAA J. Autom. Sin.* **2023**, *10*, 1647–1667. [[CrossRef](#)]
12. Augusto, A.; Belchior, R.; Correia, M.; Vasconcelos, A.; Zhang, L.; Hardjono, T. SoK: Security and Privacy of Blockchain Interoperability. In Proceedings of the 45th IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 20–23 May 2024; pp. 3840–3865.
13. Lee, S.-S.; Murashkin, A.; Derka, M.; Gorzny, J. SoK: Not Quite Water Under the Bridge: Review of Cross-Chain Bridge Hacks. In Proceedings of the IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Dubai, United Arab Emirates, 1–5 May 2023; pp. 1–14.
14. Notland, J.S.; Li, J.; Nowostawski, M.; Haro, P.H. SoK: Cross-chain bridging architectural design flaws and mitigations. *Blockchain Res. Appl.* **2026**, *7*, 100315. [[CrossRef](#)]
15. Zhou, Z.; Luo, X.; Ji, X.; Mao, J.; He, T.; Wang, J.; Wu, Q. BridgeGuard: Checking External Interaction Vulnerabilities in Cross-Chain Bridge Router Contracts Based on Symbolic Dataflow Analysis. *IEEE Trans. Dependable Secur. Comput.* **2025**, *22*, 5798–5812. [[CrossRef](#)]
16. Lin, D.; Wu, J.; Su, Y.; Zheng, Z.; Nan, Y.; Zhang, Q.; Song, B.; Zheng, Z. Connector: Enhancing the Traceability of Decentralized Bridge Applications via Automatic Cross-Chain Transaction Association. *IEEE Trans. Inf. Forensics Secur.* **2025**, *20*, 7588–7601. [[CrossRef](#)]
17. Hu, X.; Ling, Y.; Hua, J.; Dong, Z.; Sun, Y.; Qi, J. A Blockchain Cross-Chain Transaction Method Based on Decentralized Dynamic Reputation Value Assessment. *IEEE Trans. Netw. Serv. Manag.* **2024**, *21*, 5597–5612. [[CrossRef](#)]
18. Gu, C.; Ma, B.; Hu, D. A Dependable and Efficient Decentralized Trust Management System Based on Consortium Blockchain for Intelligent Transportation Systems. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 19430–19443. [[CrossRef](#)]
19. Masango, N.C.; Agushaka, J.O.; Amaefule, M.C.; Taiwo, O.; Zhang, P.; Ezugwu, A.E.; Saleem, K.; Smerat, A.; Abualigah, L. Secure and efficient cloudlet networks: Blockchain integration with agent-based proof of trust mechanism. *EURASIP J. Wirel. Commun. Netw.* **2025**, *2025*, 15. [[CrossRef](#)]
20. Shao, S.; Chen, F.; Xiao, X.; Gu, W.; Lu, Y.; Wang, S.; Tang, W.; Liu, S.; Wu, F.; He, J. IBE-BCIoT: An IBE based cross-chain communication mechanism of blockchain in IoT. *World Wide Web* **2021**, *24*, 1665–1690. [[CrossRef](#)]
21. Liu, L.; Wang, Q.; Turnbull, S.J.; Omote, K. The Validator’s Dilemma in PoW Blockchain: An Evolutionary Game Perspective. In *IEEE International Conference on Blockchain (Blockchain)*, Danzhou, China, 17–24 December 2023; IEEE: Piscataway, NJ, USA, 2023.
22. Sun, J.; Zhao, R.; Yin, H.; Cai, W. Incentive Mechanism for Redactable Blockchain Governance: An Evolutionary Game Approach. *IEEE Trans. Comput. Soc. Syst.* **2024**, *11*, 6953–6965. [[CrossRef](#)]
23. Zhang, J.; Yang, J. Blockchain-Enabled Data Supply Chain Governance: An Evolutionary Game Model Based on Prospect Theory. *Mathematics* **2026**, *14*, 432. [[CrossRef](#)]
24. Li, B.; Xiao, H. Evolutionary Game and Simulation Analysis of Supply Chain Finance Based on Blockchain Technology. In Proceedings of the 11th International Symposium on Project Management (ISPM2023), Beijing, China, 16–17 June 2023; pp. 512–524.
25. Belchior, R.; Vasconcelos, A.; Guerreiro, S.; Correia, M. A survey on blockchain interoperability: Past, present, and future trends. *ACM Comput. Surv.* **2021**, *54*, 1–41. [[CrossRef](#)]
26. Pan, C.; Liu, Z.; Liu, Z.; Long, Y. Research on Scalability of Blockchain Technology: Problems and Methods. *J. Comput. Res. Dev.* **2018**, *55*, 2099–2110.
27. Suryani, V.; Sulistyono, S. The Detection of On-Off Attacks for the Internet of Things Objects. In Proceedings of the 2018 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC), Bandung, Indonesia, 5–7 December 2018; pp. 1–5.
28. Smith, J.M. Evolution and the Theory of Games. In *Did Darwin Get It Right?* Chapman & Hall: London, UK, 1982; pp. 202–215.

29. Cheng, L.; Peng, P.; Huang, P.; Zhang, M.; Meng, X.; Lu, W. Leveraging evolutionary game theory for cleaner production: Strategic insights for sustainable energy markets, electric vehicles, and carbon trading. *J. Clean. Prod.* **2025**, *512*, 145682. [[CrossRef](#)]
30. Kader, M.S.F.; Hossain, M.M.; Ghose, S.; Zafreen, K.R. *Selection of Better Strategy for Self Organized Data Aggregation Techniques using Evolutionary Game Theory*; IEEE: Piscataway, NJ, USA, 2017.
31. Lv, Y.; Feng, R.; Ma, M.; Zhu, M.; Wu, H.; Li, X. Reinventing Multi-User Authentication Security from Cross-Chain Perspective. *IEEE Trans. Inf. Forensics Secur.* **2024**, *19*, 8908–8923. [[CrossRef](#)]
32. Lu, Z.; Wang, L.; Zhang, Z.; Huang, M.; Wang, J.; Li, M. TMT-FL: Enabling Trustworthy Model Training of Federated Learning with Malicious Participants. *IEEE Trans. Dependable Secur. Comput.* **2025**, *22*, 2723–2740. [[CrossRef](#)]
33. Josang, A.; Ismail, R. The beta reputation system. In Proceedings of the 15th Bled Conference on Electronic Commerce, Bled, Slovenia, 17–19 June 2002; pp. 324–337.

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.