2025 7th Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS) | 979-8-3315-5982-3/25/$31.00 ©2025 IEEE | DOI: 10.1109/BRAINS67003.2025.11302925

STARVESPAM: Mitigating Spam with Local Reputation in Permissionless Blockchains

Rowdy Chotkan, Bulat Nasrulin, Jérémie Decouchant, Johan Pouwelse
{R.M.Chotkan-1, B.Nasrulin, J.Decouchant, J.A.Pouwelse}@tudelft.nl
Delft University of Technology, The Netherlands

***Abstract*—Spam poses a growing threat to blockchain networks. Adversaries can easily create multiple accounts to flood transaction pools, inflating fees and degrading service quality. Existing defenses against spam, such as fee markets and staking requirements, primarily rely on economic deterrence, which fails to distinguish between malicious and legitimate users and often exclude low-value but honest activity. To address these shortcomings, we present STARVESPAM, a decentralized reputation-based protocol that mitigates spam by operating at the transaction relay layer. STARVESPAM combines local behavior tracking, peer scoring, and adaptive rate-limiting to suppress abusive actors, without requiring global consensus, protocol changes, or trusted infrastructure. We evaluate STARVESPAM using real Ethereum data from a major NFT spam event and show that it outperforms existing fee-based and rule-based defenses, allowing each node to block over 95% of spam while dropping just 3% of honest traffic, and reducing the fraction of the network exposed to spam by 85% compared to existing rule-based methods. STARVESPAM offers a scalable and deployable alternative to traditional spam defenses, paving the way toward more resilient and equitable blockchain infrastructure.**

***Index Terms*—blockchain, spam mitigation, decentralized reputation, peer-to-peer networks, Sybil resistance**

I. INTRODUCTION

The Internet has been plagued by spam for nearly half a century. In 1978, an unsolicited advertisement for a DEC mainframe was sent to hundreds of ARPANET users, now considered the first recorded case of email spam [1], [2]. Nearly five decades later, spam at the various layers of digital systems continues to impact their performance. Despite substantial advancements in filtering techniques, platform-level moderation, and machine learning, spam remains a persistent and evolving threat [3]. In 2024 alone, over 48% of global email traffic was classified as spam [4]. Centralized platforms, such as Gmail, Facebook, and Twitter, have developed powerful anti-spam tools but rely heavily on central control, content surveillance, and user profiling [1], [5]. In decentralized systems, however, spam remains a fundamental and unresolved challenge.

Because they do not benefit from centralized moderators or global coordination, decentralized systems struggle to contain spam. In particular, in blockchain networks, spam takes on new forms with broader implications. Unlike email spam, which primarily targets end-users, blockchain spam consumes shared global resources such as block space, network bandwidth, and node computation. Protocols such as Bitcoin, Ethereum, and Solana must defend against spam-like behavior that clogs mempools, inflates fees, and degrades network responsiveness.

In April 2022, Solana suffered a major outage when spam bots submitted over 4 million NFT mint transactions per second, saturating validator bandwidth and memory, and halting the chain for seven hours [6], [7]. This event marked the beginning of a period of recurrent congestion. Subsequent incidents saw over 70% of non-vote transactions fail, often due to spam bots flooding the mempool during NFT mints and speculative token launches [8], [9]. In Bitcoin, spam campaigns have included 'dust attacks' [10], where attackers send thousands of small outputs to wallets, bloating the UTXO set. In 2015, an attack filled blocks with junk outputs and consumed over 200 BTC in fees to delay transactions [11]. Ethereum has experienced similar disruptions, including a spam attack in 2016 that exploited opcodes to exhaust resources [12].

The dominant mitigation strategy in blockchain is economic deterrence. Protocols rely on various mechanisms to disincentivize abuse, such as fixed and dynamic minimum fees [13], fee auctions (e.g., EIP-1559 [14]), and staking-based access control systems [15]. While these approaches are practical against large-scale flooding, they suffer from two significant flaws. First, they fail to distinguish user intent: honest participants are penalized along with attackers during periods of congestion. Second, determined attackers can still outbid legitimate users, as evidenced by gas bidding wars during high-demand events such as NFT mints and airdrops [16].

These limitations arise from a deeper structural asymmetry in blockchain systems: creating a new identity is cheap, and costs are incurred only at execution time. Without persistent accountability or reputation, adversaries can cheaply generate new addresses or contracts to sustain flooding attacks. While economic deterrence offers a scalable defense, its side effects, particularly the exclusion of low-value but legitimate transactions, pose barriers to accessibility and fairness, especially for users operating at the network's margins. This points to a broader issue: spam in decentralized systems is not merely an economic nuisance, but a *reputation problem*.

In traditional systems, reputation plays a central role in spam mitigation. Email servers use sender reputation and content-based heuristics (e.g., SPF, DKIM, and spam filters) to block abusive accounts [3]. Social media platforms track behavioral signals (e.g., account age, follower networks, prior reports) to suppress low-quality or abusive actors [1], [5]. Even the Internet infrastructure relies on domain reputation

This work was funded by NWO/TKI grant BLOCK.2019.004.

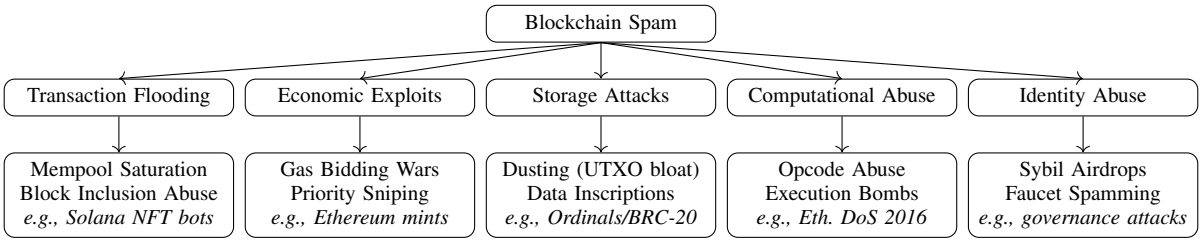

Fig. 1: Taxonomy of blockchain spam attacks by resource exhaustion or unfair access.

and IP blacklists (e.g., DNSBLs [17]) to combat spam across networks [18]. These defenses all depend on long-lived identity: misbehavior is remembered, and repeat offenders are penalized. In contrast, blockchains lack any notion of such continuity. There is no memory of past misbehavior, no account scoring, and no way to distinguish reliable senders from new or malicious ones. Each transaction is treated in isolation, allowing spammers to bypass filtering simply by rotating addresses. This absence of reputation memory makes spam particularly hard to contain in decentralized environments.

To address these issues, we present STARVESPAM, a decentralized, reputation-based spam mitigation mechanism. Our approach combines long-lived identities with a Sybil-tolerant reputation system, allowing each node to locally assess peer behavior over time and respond accordingly. Rather than filtering individual transactions or enforcing global bans, our mechanism applies subjective and adaptive rate-limiting, preserving decentralization while curbing abuse. STARVESPAM offers a novel direction for decentralized spam resistance, moving beyond fees and toward accountability without centralization.

As a summary, this work makes the following contributions:
• We propose a taxonomy of transaction-layer spam attacks, organizing common abuse patterns by the resources they exhaust or mechanisms they exploit.
• We design and prototype STARVESPAM, a decentralized spam mitigation protocol that operates at the transaction relay layer using local reputation scores and adaptive rate-limiting.
• We evaluate STARVESPAM using real-world Ethereum data from a large-scale NFT spam event, as well as synthetic scenarios with varying traffic and attacker behavior.

II. PROBLEM DESCRIPTION

Spam in blockchain systems refers to the injection of transactions that serve little or no economic purpose but consume shared network resources such as bandwidth, block space, and validator computational resources. These transactions are often submitted in high volumes to degrade network responsiveness, drive up fees, or crowd out legitimate users. While spam has been extensively studied in contexts such as email [1] and peer-to-peer overlays [19], [20], it poses unique challenges in permissionless blockchain networks. In contrast to permissioned systems, where participants can be vetted or rate-limited at the network level, permissionless protocols, such as Ethereum, Solana, and Bitcoin, must process all syntactically valid, fee-paying transactions, regardless of the sender's identity or behavior. This makes spam not only a nuisance but a recurring attack vector for denial-of-service, fee manipulation, and state bloat.

A. Taxonomy of Blockchain Spam

Spam in blockchain systems can be classified along several dimensions, including attack surface, exploited resource, and adversarial intent. Prior taxonomies have examined blockchain attacks broadly, ranging from consensus faults to application-layer exploits (e.g., [21]–[23]), emphasizing the importance of organizing threats by their system impact. Building on this perspective, we introduce a new taxonomy of spam-specific attacks, grounded in real-world observations. We manually classified attack types by reviewing post-mortems and mapping them to system-level impact (e.g., fee inflation, state bloat, or validator DoS). Fig. 1 categorizes these attacks by the primary resource they exhaust or the mechanism they exploit:

- **Transaction Flooding.** Attackers flood the network with transactions to saturate mempools or fill blocks, delaying honest activity and triggering fee spikes. For example, during Solana's 2022 NFT minting event, bots submitted over 4 million transactions per second, causing a multi-hour outage [6], [7].
- **Economic Exploits.** Spam can manipulate fee mechanisms (e.g., gas wars or priority sniping [16]) to crowd out honest users, especially during high-demand events, where well-funded spammers outbid others and undermine fairness.
- **Storage Attacks.** Techniques such as dust attacks, NFT airdrop spam, and data inscriptions (e.g., Ordinals or BRC-20 [24]) bloat the state or UTXO set, increasing long-term storage overhead for nodes.
- **Computational Abuse.** Spam can also target validator resources, as in Ethereum's 2016 DoS attack, which exploited underpriced opcodes to exhaust CPU and memory resources [12].
- **Identity-Based Exploits.** Spammers leverage disposable pseudonyms to farm airdrops, drain faucet funds, and influence votes [25]. These Sybil-style attacks degrade incentive alignment in user-driven protocols.

While diverse in mechanism and impact, these spam attacks exploit a common structural weakness: the lack of long-term accountability in permissionless systems.

TABLE I: Comparison of spam mitigation approaches in blockchain systems

Approach	Decentralized	Sybil-Tolerant	Behavior-Aware	Tx-Level	Protocol Changes
Fee Thresholds (EIP-1559, Solana)	✓	–	–	✓	–
Static Heuristics (e.g., BanMan)	✓	–	∼	✓	✓
Reputation Filters (Zhang et al.)	✓	∼	✓	✓	–
Identity Systems (BrightID, Gitcoin)	∼	✓	–	–	–
StarveSpam (Ours)	✓	✓	✓	✓	–

B. Why Spam Persists

The persistence of spam in decentralized networks stems from two root causes:

- **Identities are cheap and disposable**: Participants can generate new addresses at no cost. There is no inherent notion of continuity, accountability, or history tied to a participant.
- **Protocols are behavior-agnostic**: Block producers accept any valid transaction that pays a fee, treating spammers and legitimate users alike.

At their core, most blockchain protocols are stateless with respect to participant behavior. This neutrality may uphold decentralization, but it also enables attackers to repeat abuse without consequence. These properties make blockchains particularly vulnerable to Sybil attacks, as adversaries can flood the system under many pseudonyms without consequences.

Limitations of Economic Defenses: Current anti-spam mechanisms primarily rely on economic deterrence: minimum fees, dynamic fee markets (e.g., EIP-1559 [14]), and stake-based access control. These tools increase the cost of spamming but suffer from two key limitations:

- **Collateral damage.** Fee pressure penalizes honest and malicious users alike, without regard for intent or value.
- **Insufficient deterrence.** Motivated attackers can outbid others, especially when financial rewards (e.g., airdrops or MEV [26]) outweigh the costs.

These defenses fail not because fees are ineffective, but because they ignore identity and behavioral history. Without the ability to distinguish trustworthy behavior from manipulation, protocols remain vulnerable to well-funded adversaries.

The Need for Behavioral Accountability: These challenges highlight a fundamental gap: decentralized systems currently lack mechanisms for associating actions with persistent identity or memory. Without some form of behavioral accountability, protocols remain vulnerable to spam by rational or malicious actors who exploit protocol neutrality. To address this, we seek a decentralized approach that allows networks to retain, distinguish, and adapt to participant behavior, without compromising decentralization or permissionlessness.

III. Related Work

Spam resistance in decentralized networks has been approached from several angles, including fee mechanisms, propagation heuristics, reputation systems, and cryptographic rate-limiting. We discuss these mechanisms in the following.

A. Economic Spam Deterrents

Most blockchains rely on cost barriers to discourage spam. Bitcoin enforces minimum relay fees and dust limits [27], while Ethereum introduced EIP-1559 to dynamically adjust base fees under congestion [14]. IOTA requires Proof-of-Work for transaction submission [13], and Solana supports priority fees to bid for inclusion. EOS and Internet Computer (ICP) tie transaction capacity to staked system resources (e.g., CPU, RAM) [28], [29].

These mechanisms make large-scale spam costly, but do not differentiate intent or reputation. During high-demand periods (e.g., NFT mints), honest users may still be priced out by well-funded attackers [16]. Moreover, spam campaigns, such as Bitcoin's 2015 stress test, demonstrated that adversaries may willingly spend hundreds of BTC to degrade service [11].

B. Protocol-Level Heuristics

Several blockchain clients use implementation-specific spam filtering. Bitcoin's *BanMan* module tracks misbehaving peers (e.g., invalid blocks or headers) and discourages them by omitting them from peer discovery [30]. However, bans are local, temporary, and not persistent across restarts. Ethereum has used opcode repricing to mitigate underpriced computational attacks such as the 2016 DoS campaign [12], and Solana is exploring localized fee markets per smart contract [31].

These defenses are often reactive and fragile. They require tuning, depend on protocol changes, and do not generalize across ecosystems.

C. Reputation and Identity Mechanisms

Reputation systems have been deployed in peer-to-peer overlays such as Credence [32], EigenTrust [33], and MeritRank [34] to assess peer quality. Zhang *et al.* [35] propose a decentralized reputation mechanism for Bitcoin that rates peers based on the validity and usefulness of forwarded transactions. These systems typically assume strong identities and consistent peer interaction, which may not hold in all blockchain environments. Unlike this work, we explicitly embrace the assumption of persistent pseudonyms and propose a Sybil-tolerant design that uses local and behavior-based reputation for spam mitigation.

Projects such as BrightID, Gitcoin Passport, and Proof of Humanity[1] attempt to create Sybil-resistant identities, but are typically deployed at the application layer rather than for transaction filtering. Spam resistance through identity in public blockchains remains underexplored.

[1]https://brightid.org; https://gitcoin.co/passport; https://proofofhumanity.id

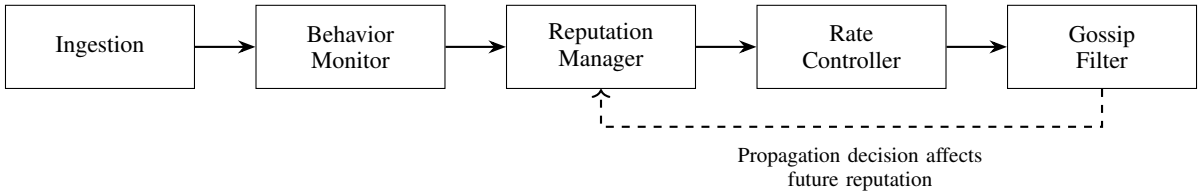

Fig. 2: Overview of STARVESPAM's transaction pipeline.

D. Privacy-Preserving and Challenge-Response Defenses

Unlinkability and challenge-response have been used to rate-limit abuse without full identity. Inselvini *et al.* [36] propose using ZK-SNARKs for anonymous, spam-resistant content sharing with centralized or federated gatekeepers. CAPTCHA and challenge-response methods like XEP-0377 [37] allow user-driven blocking and spam reporting in XMPP-based systems. These approaches rely on user friction or central authorities and are not well-suited to low-latency, decentralized transaction propagation.

While prior work covers fee tuning, heuristics, and identity systems, each falls short in one or more dimensions, such as Sybil tolerance or behavioral awareness. Table I summarizes these trade-offs. In contrast, our approach combines long-lived pseudonyms with local, behavior-based reputation in a fully decentralized setting. It applies adaptive rate-limiting without global state, identity uniqueness, or centralized enforcement.

IV. SYSTEM DESIGN

We design STARVESPAM for permissionless blockchain networks, where nodes communicate over a peer-to-peer overlay and autonomously decide which transactions to accept, relay, or drop. Identities are cheap and ephemeral: nodes are identified by public keys and may generate pseudonyms at negligible cost. The network lacks central coordinators, shared state, or global reputation; each node relies solely on its own observations to assess peer behavior.

In this setting, we assume that adversaries may control many Sybil identities and generate large volumes of low-utility transactions. At the same time, honest nodes behave consistently and can build local trust over time. STARVESPAM exploits this asymmetry: it enables nodes to classify transactions based on behavioral signals, assign local reputation scores to senders, and apply adaptive rate-limiting, all without protocol changes or global coordination. STARVESPAM operates entirely at the transaction relay layer. As illustrated in Fig. 2, it consists of modular components that allow each node to assess incoming transactions independently and degrade service for abusive senders, while preserving throughput and fairness for legitimate participants.

A. Design Rationale

STARVESPAM is structured as a pipeline of modular components, each serving a distinct role in the local spam mitigation process. The pipeline reflects a logical dependency: transaction ingestion must precede behavioral classification, which in turn feeds reputation scoring, rate control, and gossip filtering. This modular design allows each node to make independent relay decisions while retaining the ability to adapt and tune specific components. We selected these components based on three criteria: (i) their feasibility in real-world node implementations; (ii) their alignment with observed spam behaviors; and (iii) their synergy in preserving decentralization while enabling behavioral accountability.

B. Transaction Ingestion

When a transaction is received from a peer, it first passes through the transaction ingestion layer, which extracts relevant metadata and routes it to downstream modules. Specifically, the ingestion layer parses the sender's public key (from the transaction signature), the gas price or fee information, the calldata size and content hash, and the touched contract addresses or UTXOs (depending on the blockchain).

In permissionless networks like Ethereum or Solana, senders are identified by public keys, which are cheap to generate. While this prevents strong identity, we assume that most spam campaigns operate under persistent pseudonyms (e.g., fixed wallets or contract deployers) due to cost structures (staking, airdrop eligibility, access to private keys). This assumption is consistent with prior work that shows that long-running spam campaigns reuse addresses for operational or strategic reasons [11].

The ingestion layer does not perform any filtering itself, but prepares transactions for classification by computing per-transaction features used by the behavior monitor. It also maintains a rolling history of recent transactions from each sender to support local temporal analysis (e.g., rate tracking and inclusion ratios). The history is stored as a fixed-size queue of the last N transactions or covering a time window of ΔT seconds, whichever is longer. In our prototype (see section V), we use $N = 20$ and $\Delta T = 60$.

C. Behavior Monitoring

The behavior monitor evaluates incoming transactions using a set of interpretable, rule-based heuristics. Its purpose is to detect abuse patterns, such as flooding, resource exhaustion, or data duplication, without relying on centralized labeling or black-box classifiers. Inspired by documented spam campaigns across Ethereum, Solana, and Bitcoin, we design the monitor around five transaction-level features commonly associated with abusive behavior:

- **Low gas price or fee.** The transaction offers a gas price below the 25th percentile of recent activity, reflecting low-fee spam tactics used in attacks like Bitcoin's 2015 stress tests and Ethereum's gas wars [11], [38].
- **Redundant calldata.** The calldata or payload matches a recent transaction from the same sender, capturing duplication patterns seen in NFT mint spam and contract call floods [8].
- **High revert rate.** The sender shows a high rate of reverted transactions (e.g., 4 out of the last 10), indicating abusive behavior. Revert-heavy spam is common in DeFi bot floods and failed arbitrage attacks, especially on Solana [9], [35].
- **Burst frequency.** The sender has submitted more than k transactions in a short time window (e.g., 5 in 3 seconds), a pattern typical of denial-of-service and congestion attacks across major blockchains [7], [11].
- **Non-inclusion penalty.** A large share of the sender's recent transactions remain unconfirmed after β blocks (e.g., 7 of 10 pending after 20 blocks), indicating low economic utility. This heuristic captures low-impact activity that bloats mempools without affecting state [35], [39].

Each rule is lightweight and implementable using local node observations. Nodes may assign weights to each condition to compute a spam score or apply simple logic (e.g., flag as spam if ≥ 2 conditions are met, as applied in our experiments). The classification result is then forwarded to the reputation manager to update the sender's long-term score.

This rule-based approach strikes a balance between robustness and explainability. It avoids the operational overhead of machine learning, while remaining adaptable; thresholds can be tuned, and new rules can be added as spam tactics evolve. Each feature is grounded in real-world attack patterns, ensuring both defensibility and empirical relevance.

D. Reputation Management

The reputation manager maintains a local, persistent score $r_i \in [0, 1]$ for each sender, based on classifications from the behavior monitor. This score serves as a proxy for trustworthiness, informing transaction admission and propagation decisions. In STARVESPAM, reputation is strictly local and subjective: each node scores addresses independently, using only its own observations.

Each sender's score is updated using an exponentially weighted moving average (EWMA) of past classification outcomes, balancing responsiveness to recent behavior with resistance to noise. Transactions labeled as spam decrement the score, while those classified as benign increment it. The update function includes:

- **Score decay.** To allow forgiveness over time, scores gradually return to neutral if no further spam is observed.
- **Recency bias.** Recent transactions are weighted more heavily than older ones, enabling the system to respond quickly to changes in behavior.
- **Capping and clipping.** Scores are bounded within a fixed range (i.e., $[0, 1]$) to simplify downstream rate control and filtering logic.

This design is inspired by reputation systems in peer-to-peer overlays (e.g., EigenTrust [33], Credence [32], MeritRank [34]), and adapted for pseudonymous blockchain contexts. Since identities are derived from public keys and can be easily regenerated, STARVESPAM is designed to be Sybil-tolerant:

- Fresh identities with little history begin with a neutral reputation and are conservatively throttled until they demonstrate reliability.
- Rapid identity rotation provides no advantage, as spammers cannot accumulate reputation across short-lived addresses.
- Long-lived, honest participants gradually build a stronger reputation, enabling higher throughput and improved responsiveness.

This approach punishes repeat abusers while allowing recovery and resilience to misclassification. It avoids the brittleness of centralized blocklists and ensures that honest users, especially those with low activity or intermittent connectivity, are not permanently excluded. However, because reputation builds over time, new or infrequent participants may be conservatively throttled at first—a necessary trade-off to limit Sybil abuse. Nodes can tune initial scores or thresholds to better balance openness and caution.

E. Rate Control and Admission Policy

Once a sender's reputation score is updated, the rate controller determines how to handle subsequent transactions. This module enforces adaptive, local rate-limiting, prioritizing resources such as mempool space, bandwidth, and validation cycles for reputable participants, while progressively throttling low-reputation identities. We partition reputation scores into three regions to guide rate control decisions. For reference, in our evaluation (see section V) nodes with scores above $\tau_{\text{high}} = 0.8$ are considered high-reputation, those below $\tau_{\text{low}} = 0.2$ are low-reputation, and scores in between are treated as moderate-reputation. Each incoming transaction is processed based on the sender's current reputation r_i:

- **High-reputation**—Immediate mempool admission; applies when $r_i \geq \tau_{\text{high}}$.
- **Moderate-reputation**—Subject to queuing or delay, especially under congestion; applies when $\tau_{\text{low}} \leq r_i < \tau_{\text{high}}$.
- **Low-reputation**—Deprioritized or dropped entirely under load, effectively starved of access until reputation improves; applies when $r_i < \tau_{\text{low}}$.

This design acts as a soft admission gate: it does not impose rigid bans, but degrades service quality in proportion to observed behavior. Unlike fixed fee thresholds or static opcode bans, rate control offers gradual, feedback-driven enforcement that balances fairness with responsiveness. The controller also adapts to system conditions. Under normal load, nodes may accept more low-reputation traffic to preserve openness.

Under stress, thresholds tighten to prioritize trusted flows. This elasticity mirrors fee market dynamics but is guided by reputation rather than gas price.

By mediating between the behavior monitor and the mempool, the rate controller ensures that abusive identities cannot monopolize resources, even when they adhere to fee policies. The rate controller is the mechanism through which reputation directly shapes transaction acceptance.

F. Gossip Filtering

In addition to controlling local mempool admission, STARVESPAM influences how transactions are propagated across the network. The gossip filter leverages sender reputation to suppress the spread of spam transactions, reducing their visibility and limiting the chance they will reach validators or block producers via other peers. In most blockchain P2P layers (e.g., Ethereum's devp2p or Solana's Turbine), transactions are gossiped opportunistically to random or stake-weighted peers. By default, nodes forward transactions indiscriminately, consuming unnecessary bandwidth and memory during spam events. STARVESPAM modifies this behavior by broadcasting transactions from high-reputation senders without restriction, while treating low-reputation traffic more conservatively: such transactions may be forwarded with reduced priority, sampled at a lower rate, or dropped under congestion or buffer pressure.

This mechanism ensures that even if one permissive peer accepts a spam transaction, it is unlikely to propagate widely, effectively containing its blast radius. Because reputation is local, each node may make different forwarding decisions, creating a form of subjective resistance that is hard to game. Spammers must rebuild trust independently with every peer they contact. This is particularly valuable in settings where spam is economically rational (e.g., Solana bots or arbitrage attacks), as it introduces non-fee-based pressure: spammers may continue paying for bandwidth, but their reach is throttled by network-level filtering. Additionally, the gossip filter acts as a natural reputation sink, progressively isolating abusive addresses and limiting their ability to amplify attacks through peer propagation.

G. Deployment and Integration

STARVESPAM is modular, decentralized, and incrementally deployable. It operates entirely at the transaction relay layer and requires no changes to consensus, transaction formats, or global coordination. As such, it can be integrated into a wide range of blockchain clients and overlays, without disrupting core protocol functionality (e.g., Ethereum, Bitcoin, and L2 systems). Nodes run it independently as a local pre-processing module that filters, scores, and regulates incoming transactions. Since all components (classification, reputation tracking, rate limiting, gossip filtering) rely solely on local observations and subjective policies, there is no need to trust external inputs or converge on a global view. STARVESPAM is therefore well-suited for permissionless, pseudonymous networks.

The system can be integrated at various points in the stack: full nodes use it to reduce mempool and P2P load, validators or sequencers apply reputation-aware admission to protect block production, light clients avoid acting as spam amplifiers, and rollups filter spam before costly settlement operations.

Rate control is local and modular. Nodes can tune parameters such as thresholds, decay rates, and queue sizes based on hardware constraints, risk tolerance, or network role, thereby calibrating trade-offs between openness and abuse resistance. Its modular architecture supports upgrades to detection logic or reputation models without coordination or hard forks. While the current design uses rule-based heuristics, the framework is extensible. Nodes may incorporate local machine learning models or integrate signals from off-chain reputation and identity systems. The core rate-control and reputation logic remains compatible with such enhancements.

V. EXPERIMENTAL EVALUATION

We evaluate STARVESPAM through three experiments, each targeting a key aspect of decentralized spam mitigation. First, we replay a real-world spam event to assess filtering under short-term load. Second, we simulate a large-scale network to observe how local reputations evolve for honest, malicious, and oscillating nodes. Third, we model peer-to-peer gossip to evaluate how local filtering limits network-wide spam propagation.

A. Setup and Dataset

All experiments run in a custom discrete-event simulator built with SimPy, where each node independently applies local admission policies. We use both real-world Ethereum data and synthetic traces, measuring both per-node and network-wide outcomes under varied traffic and behavioral scenarios.

Our first and third experiments use a dataset of $50{,}000$ Ethereum transaction data from the Otherside NFT mint (April 30, 2022), an event that caused extreme congestion and over $150 million in gas fees, with many transactions failing due to mempool flooding [40], [41]. The dataset comprises all public transactions during the event window, sourced from the Ethereum network, and covers the fields listed in Tab. II. Our dataset includes only transactions that were eventually confirmed on-chain, omitting those dropped or delayed indefinitely. While this excludes some aggressive spam attempts, we capture a broad range of low-utility and reverted transactions representative of real-world abuse. The second experiment uses synthetic traces that simulate up to 100 nodes, with scheduled patterns such as diurnal activity, spam bursts, and behavioral shifts.

TABLE II: Transaction fields used in the dataset

Field	Description
`timestamp`	The Unix timestamp of the transaction.
`from_address`	Ethereum address that initiated the transaction.
`calldata`	Encoded input data for contract execution.
`gas_price`	Gas price offered by the sende, in Gwei.
`receipt_status`	Execution status (1 = success, 0 = revert).
`receipt_gas_used`	Amount of gas consumed during execution.

To assign ground-truth labels, we apply a multi-heuristic classifier based on established indicators. A transaction is labeled as spam if it satisfies at least two of five such heuristics (detailed in Tab. III). The classifier operates offline with fixed thresholds and relies solely on general, transaction-level features. While some features overlap with those used by STARVESPAM's behavior monitor, the labeling process is entirely decoupled, having no access to its internal logic. The classifier uses static rules to assign evaluation labels, whereas STARVESPAM updates peer reputation over time and applies adaptive rate control. This separation avoids circularity and ensures that the system is not tuned to match the labeling strategy. Approximately 62% of transactions are flagged as spam, consistent with post-mortems that report widespread failure and abuse [42].

B. Filtering Effectiveness under Spam Load

In our first experiment, we measure how effectively STARVESPAM filters spam while preserving honest participation during a high-volume, adversarial workload. We replay the labeled transaction trace from the Otherside NFT mint and compare STARVESPAM against five baselines:

- **Naive**: accepts all transactions unconditionally.
- **Fee filter**: drops transactions below the 10th percentile of gas price.
- **BanMan**: penalizes peers based on low-fee and reverted transactions.
- **EIP-1559**: drops transactions below a dynamic base fee (modeled as the 25th percentile gas price over a rolling window).
- **SIMD-110**: simulates per-account congestion penalties inspired by Solana's proposed write-lock markets.

Fig. 3 shows the number of spam transactions accepted (false negatives) and honest transactions dropped (false positives) under each policy after $50{,}000$ transactions. STARVESPAM achieves the best trade-off, accepting only 4.6% of spam while dropping around 3% of honest transactions. By comparison, Naive, Fee Filter, BanMan, and EIP-1559 all allow over 50% of spam through. Although SIMD-110 significantly reduces spam acceptance, it drops a much larger share of honest traffic, demonstrating poor fairness. STARVESPAM combines precise spam filtering with adaptive throttling to achieve the most optimal trade-off: it blocks the majority of spam while minimizing the loss of honest transactions.

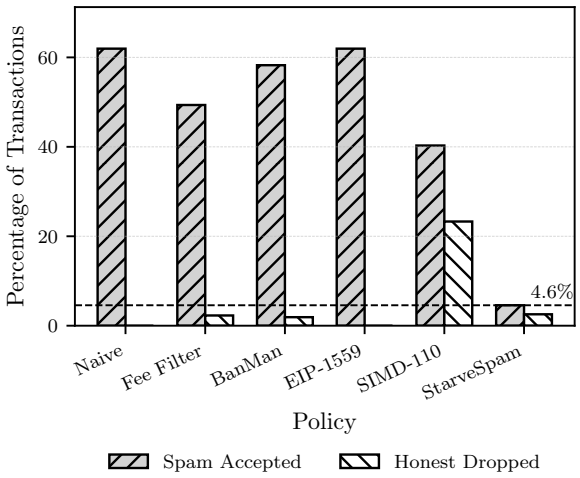

Fig. 3: Percentage of transactions accepted or dropped by each policy.

C. Reputation Evolution Over Time

This experiment evaluates STARVESPAM's ability to dynamically adjust peer reputations based on observed behavior. The goal is to assess whether the system can distinguish honest from malicious nodes and support recovery for those that improve over time.

We simulate a network of 100 peers, divided into three

TABLE III: Heuristics Used in Rule-Based Spam Classification. Each rule is derived from documented attack patterns and designed for local, explainable evaluation.

Feature	Description	Rationale
Duplicate calldata	Transaction shares its calldata with one or more others in the trace.	Large-scale spam campaigns often submit identical or near-identical transactions in bulk, aiming to exploit timing advantages during NFT drops or airdrop eligibility checks. Repetition is a strong signal of scripted behavior.
Reverted execution	Transaction fails to execute and reverts (receipt status = 0).	High revert rates indicate speculative, zero-signal activity (e.g., failing arbitrage attempts). These transactions consume compute and gas while producing no state changes, degrading system utility.
Low fee	Gas price is below the 10th percentile of all transactions in the dataset.	Spam bursts are frequently issued at minimum viable cost. Abusers exploit mispriced fee periods or attempt to sneak spam through low-priority lanes, especially during periods of low congestion.
Low complexity	Gas used by the transaction falls below the 10th percentile of the trace.	Simple, no-op, or filler transactions are characteristic of congestion attacks and spam waves. Such transactions provide little or no economic utility but burden bandwidth and processing.
Burst activity	Sender issues multiple transactions in rapid succession within a few seconds.	Spammers often launch transactions in bursts to saturate mempools or maximize the chance of inclusion. This is common in denial-of-service scenarios and competitive bidding situations.

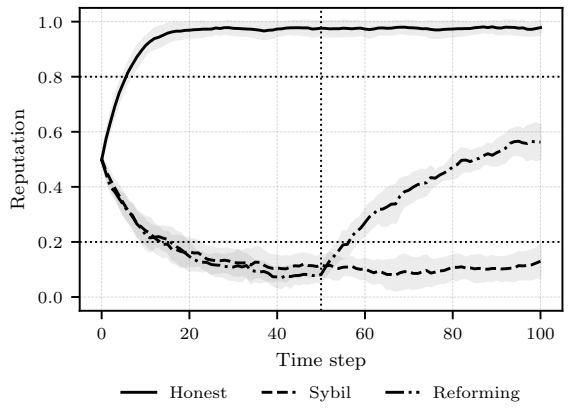

Fig. 4: Reputation trajectories for honest, Sybil, and reforming nodes.

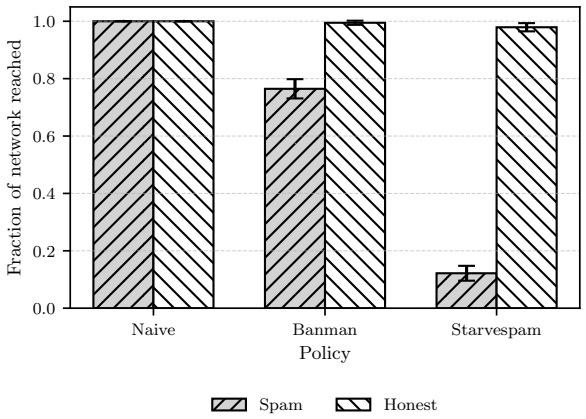

Fig. 5: Average network coverage of spam and honest transactions under different relay policies.

behavioral categories: honest, Sybil, and reforming. All nodes start with equal reputation (0.5). Honest nodes consistently relay legitimate traffic, while Sybil nodes exhibit spam-like behavior throughout the simulation. Reforming nodes behave like Sybils initially but switch to honest behavior at time step 50. Each peer's reputation is updated locally over 100 time steps using the reputation mechanism described in Section IV.

Fig. 4 shows the average reputation of each peer class over time. Honest nodes quickly converge above 0.8, while Sybil nodes degrade below 0.2. Reforming peers recover after switching behavior, demonstrating that STARVESPAM supports adaptive trust without permanent penalties. Shaded regions indicate each group's standard deviation, capturing behavioral variance. These results confirm that STARVESPAM distinguishes peer intent and maintains fairness in dynamic settings.

D. Spam Propagation Suppression

We evaluate how well each policy contains spam propagation across the network by simulating transaction relay over a randomized P2P overlay. Each transaction starts from a random node and propagates to neighbors unless filtered by local policies. We track and report the average number of nodes that each spam or honest transaction reaches under three strategies: Naive, BanMan, and STARVESPAM.

We restrict our comparison to these three policies as they represent distinct and relevant points in the design space. Naive flooding models the absence of filtering, providing an upper bound on reachability. BanMan reflects Bitcoin's in-protocol peer discouragement mechanism, applying local penalties to misbehaving senders based on fixed rules. STARVESPAM, in contrast, uses decentralized, adaptive reputation to enforce propagation limits. We omit global rate-limiting, centralized moderation, or application-layer filters, as these are incompatible with the decentralized, relay-layer setting we target.

Fig. 5 shows the results. Under the Naive policy, both spam and honest transactions reach the entire network, as expected in unrestricted flooding. BanMan achieves modest suppression, blocking some spam at intermediate hops, but still allows broad propagation. In contrast, STARVESPAM sharply reduces spam reach, with most spam transactions failing to spread beyond a small subset of peers. Meanwhile, honest transactions continue to propagate effectively, nearly matching the coverage of the Naive baseline.

VI. CONCLUSION

Spam remains a persistent and costly challenge in blockchain networks, where permissionless access and cheap identities allow adversaries to degrade availability, inflate fees, and disrupt services. Existing defenses, primarily economic deterrence or protocol-specific heuristics, struggle to balance openness with resistance to abuse, often penalizing legitimate users or failing to suppress coordinated attacks.

We introduced STARVESPAM, a decentralized, reputation-based spam mitigation system designed for the transaction relay layer. By leveraging local observations of peer behavior, STARVESPAM enables nodes to adaptively throttle abusive senders without requiring global coordination, protocol changes, or centralized infrastructure. This addresses a key gap in existing stateless, identity-agnostic protocols. Because it is fully local and modular, STARVESPAM can be incrementally adopted by relay nodes with minimal integration overhead.

Through simulations grounded in a real-world spam event, we demonstrated that STARVESPAM achieves high accuracy in suppressing spam, preserves accessibility for honest users, and significantly limits the spread of malicious transactions across the network. Compared to fee-based or rule-based baselines, STARVESPAM offers a more flexible and fair foundation for resilient spam prevention in decentralized environments.

Future work includes extending STARVESPAM to multi-chain and Layer 2 architectures, integrating privacy-preserving identity systems, and validating its design in live testnet deployments. Further directions include refining the scoring model with machine learning or game-theoretic incentive alignment, and studying deployment incentives to promote adoption across heterogeneous relayers.

REFERENCES

[1] E. Ferrara, "The history of digital spam," *Communications of the ACM*, vol. 62, no. 8, pp. 82–91, 2019.

[2] B. Templeton, "Reaction to the dec spam of 1978." [Online]. Available: https://www.templetons.com/brad/spamreact.html

[3] N. Ahmed, R. Amin, H. Aldabbas, D. Koundal, B. Alouffi, and T. Shah, "Machine learning techniques for spam detection in email and iot platforms: analysis and research challenges," *Security and Communication Networks*, vol. 2022, no. 1, p. 1862888, 2022.

[4] T. Kulikova, O. Svistunova, R. Dedenok, A. Kovtun, I. Shimko, and A. Lazaricheva, "Spam and phishing in 2024," Feb 2025. [Online]. Available: https://securelist.com/spam-and-phishing-report-2024/115536/

[5] Meta, "Transparency center," https://transparency.meta.com/, n.d., accessed: May 2025.

[6] S. Labs, "Solana network status and incident report," 2022, accessed: May 2025. [Online]. Available: https://status.solana.com

[7] D. Nelson, "Solana goes dark for 7 hours as bots swarm 'candy machine' nft minting tool," 2022, accessed: May 2025. [Online]. Available: https://www.coindesk.com/tech/2022/05/01/solana-goes-dark-for-7-hours-as-bots-swarm-candy-machine-nft-minting-tool

[8] X. Zheng, Z. Wan, D. Lo, D. Xie, and X. Yang, "Why does my transaction fail? a first look at failed transactions on the solana blockchain," *arXiv preprint arXiv:2504.18055*, 2025.

[9] Blockworks, "Solana has a spam problem. can it be fixed?" 2024, accessed: May 2025. [Online]. Available: https://blockworks.co/news/solana-spam-problem

[10] Binance Academy, "Dusting attacks explained," 2020, accessed: May 2025. [Online]. Available: https://academy.binance.com/en/articles/what-is-a-dusting-attack

[11] K. Baqer, D. Y. Huang, D. McCoy, and N. Weaver, "Stressing out: Bitcoin "stress testing"," in *Financial Cryptography and Data Security: FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, February 26, 2016, Revised Selected Papers 20*. Springer, 2016, pp. 3–18.

[12] J. Wilcke, "The ethereum network is currently undergoing a dos attack," 2016, accessed: May 2025. [Online]. Available: https://blog.ethereum.org/2016/09/22/ethereum-network-currently-undergoing-dos-attack

[13] O. Saa, A. Cullen, and L. Vigneri, "Iota 2.0 incentives and tokenomics whitepaper," 2023.

[14] V. Buterin *et al.*, "Eip-1559: Fee market change for eth 1.0 chain," 2019, ethereum Improvement Proposal. [Online]. Available: https://eips.ethereum.org/EIPS/eip-1559

[15] ——, "Ethereum white paper," *GitHub repository*, vol. 1, pp. 22–23, 2013.

[16] D. Jones, "Gas wars and failed transactions: The true cost of otherdeeds," May 2022. [Online]. Available: https://www.redlion.news/article/gas-wars-and-failed-transactions-the-true-cost-of-otherdeeds

[17] Spamhaus Project, "The spamhaus block list (sbl)," 2025, accessed: May 2025. [Online]. Available: https://www.spamhaus.org/blocklist

[18] J. Levine, "Dns blacklists and whitelists," Internet Research Task Force (IRTF), Tech. Rep., 2010, RFC 5782.

[19] K. Walsh and E. G. Sirer, "Fighting peer-to-peer spam and decoys with object reputation," in *Proceedings of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems*, 2005, pp. 138–143.

[20] E. Zhai, R. Chen, E. K. Lua, L. Zhang, H. Sun, Z. Cai, S. Qing, and Z. Chen, "Spamresist: making peer-to-peer tagging systems robust to spam," in *GLOBECOM 2009-2009 IEEE Global Telecommunications Conference*. IEEE, 2009, pp. 1–6.

[24] K. Torpey, "Ordinal inscriptions and brc-20 tokens cause bitcoin fee spike," 2023, accessed: May 2025. [Online]. Available: https://coinmarketcap.com/academy/article/ordinal-inscriptions-and-brc-20-tokens-cause-bitcoin-fee-spike

[21] Y. Chen, H. Chen, Y. Zhang, M. Han, M. Siddula, and Z. Cai, "A survey on blockchain systems: Attacks, defenses, and privacy preservation," *High-Confidence Computing*, vol. 2, no. 2, p. 100048, 2022.

[22] A. Alkhalifah, A. Ng, A. Kayes, J. Chowdhury, M. Alazab, and P. A. Watters, "A taxonomy of blockchain threats and vulnerabilities," in *Blockchain for Cybersecurity and Privacy*. CRC Press, 2020, pp. 3–28.

[23] E. Lubes and J. M. Pelletier, "A tree-mapped taxonomy of blockchain attacks," in *2023 7th Cyber Security in Networking Conference (CSNet)*. IEEE, 2023, pp. 233–237.

[25] K. Weiss, "Building sybil resistance using cost of forgery," accessed: May 2025. [Online]. Available: https://www.gitcoin.co/blog/cost-of-forgery

[26] B. Nasrulin, G. Ishmaev, J. Decouchant, and J. Pouwelse, "Lo: An accountable mempool for mev resistance," in *Proceedings of the 24th International Middleware Conference*, 2023, pp. 98–110.

[27] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[28] EOS.IO, "Eos.io technical white paper v2," 2017, accessed: May 2025. [Online]. Available: https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md

[29] D. Team *et al.*, "The internet computer for geeks," *Cryptology ePrint Archive*, 2022.

[30] B. developers, "src/banman.cpp in Bitcoin repository," GitHub repository, 2023, accessed: May 2025. [Online]. Available: https://github.com/bitcoin/bitcoin/blob/0857f2935f90df9c3d303582e5b62a9c8dedd9d7/src/banman.cpp

[31] G. Angeris, T. Diamandis, and C. Moallemi, "Multidimensional blockchain fees are (essentially) optimal," *arXiv preprint arXiv:2402.08661*, 2024.

[32] K. Walsh and E. G. Sirer, "Experience with an object reputation system for peer-to-peer filesharing." in *NSDI*, vol. 6, 2006, pp. 1–1.

[33] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in p2p networks," in *Proceedings of the 12th international conference on World Wide Web*, 2003, pp. 640–651.

[34] B. Nasrulin, G. Ishmaev, and J. Pouwelse, "Meritrank: Sybil tolerant reputation for merit-based tokenomics," in *2022 4th Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS)*. IEEE, 2022, pp. 95–102.

[35] J. Zhang, Y. Cheng, X. Deng, B. Wang, J. Xie, Y. Yang, and M. Zhang, "Preventing spread of spam transactions in blockchain by reputation," in *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*. IEEE, 2020, pp. 1–6.

[36] A. Inselvini, "Spam prevention using zk-snarks for anonymous peer-to-peer content sharing systems," *arXiv preprint arXiv:2103.02061*, 2021.

[37] S. Whited and G. Der Kinderen, "Xep-0377: Spam reporting," https://xmpp.org/extensions/xep-0377.html, 2025, accessed: May 2025.

[38] J. Lopp, "A history of bitcoin transaction dust & spam storms," 2021. [Online]. Available: https://blog.lopp.net/history-bitcoin-transaction-dust-spam-storms/

[39] B. Nasrulin, R. Chotkan, and J. Pouwelse, "Sustainable cooperation in peer-to-peer networks," in *2023 IEEE 48th Conference on Local Computer Networks (LCN)*. IEEE, 2023, pp. 1–9.

[40] E. Ongweso Jr., "Bored ape virtual land sale breaks ethereum, wastes $180 million in fees," *Vice*, 2022, accessed: May 2025. [Online]. Available: https://www.vice.com/en/article/bored-ape-virtual-land-sale-breaks-ethereum-wastes-dollar180-million-in-fees

[41] A. Hern, "Yuga labs apologises after sale of virtual land crashes ethereum," *The Guardian*, 2022. [Online]. Available: https://www.theguardian.com/technology/2022/may/02/yuga-labs-apologises-after-sale-of-virtual-land-crashes-ethereum

[42] A. Castor, "Yuga labs' otherside land sale turns into a giant gas war," 2022, accessed: May 2025. [Online]. Available: https://amycastor.com/2022/05/01/yuga-labs-otherside-land-sale-turns-into-a-giant-gas-war