https://doi.org/10.26219/heal.aueb.9401

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

SCHOOL OF INFORMATION SCIENCES & TECHNOLOGY

Master of Sciences (MSc)

IN INFORMATION SYSTEMS DEVELOPMENT & SECURITY

MSc THESIS

"Blockchain-Based Incentivization for Security Practices in IIoT-Networks"

IASON CHATZOPOULOS

p3312322

ATHENS, JANUARY 2025

https://doi.org/10.26219/heal.aueb.9401

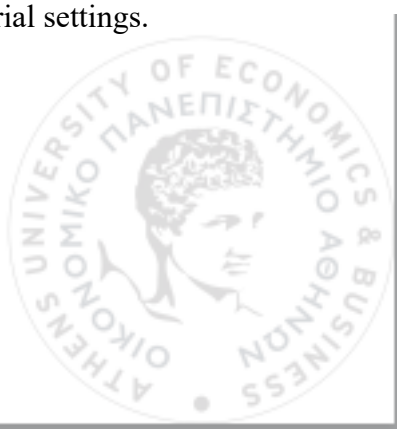
Abstract

The rapid proliferation of Industrial Internet of Things (IIoT) devices introduces significant security risks, particularly concerning unauthorized information flows and breaches of data confidentiality. Traditional access control methods often fall short in addressing dynamic, decentralized IIoT environments, necessitating more adaptive and robust security mechanisms. This thesis introduces a novel blockchain-based incentivization framework aimed at enhancing security practices within IIoT networks through decentralized, economically driven incentives.

Our proposed solution integrates Bayesian probabilistic modeling for dynamic, real-time detection of illegal information flows, employing Laplace smoothing techniques to mitigate zero-frequency issues and enhance responsiveness to emerging threats. The validated statistical results from off-chain data aggregation processes are securely transmitted via an oracle microservice implemented using Java and Spring Boot frameworks, which interacts seamlessly with Ethereum smart contracts through Web3j. Smart contracts automate the rewarding of compliant nodes and penalizing of malicious activities, employing a sophisticated ERC-20 token-based staking and slashing mechanism.

Empirical evaluations demonstrate that the Bayesian risk assessment model rapidly converges and adapts under various realistic IIoT scenarios, providing stable and accurate risk estimations. Scalability and latency analyses confirm that blockchain integration, coupled with batch processing strategies, yields acceptable performance overheads while significantly enhancing security accountability and operational resilience. Comparative analysis against existing methods such as Operation Interruption (OI) and Time-Based Operation Interruption (TBOI) protocols underscores the unique strengths of our incentive-driven approach in fostering initiative-taking compliance across large-scale IIoT deployments.

The findings presented affirm the practicality and effectiveness of blockchain-based incentivization as a complementary and scalable approach to addressing contemporary IIoT security challenges, providing clear pathways for adoption in real-world industrial settings.



https://doi.org/10.26219/heal.aueb.9401

Keywords

Blockchain, Industrial Internet of Things (IIoT), Cybersecurity, Information Flow Control, Bayesian Risk Assessment, Smart Contracts, ERC-20, Oracle Microservice, Incentivization, Decentralized Security

Greek Title

“Παροχή κινήτρων με χρήση blockchain για πρακτικές ασφάλειας σε βιομηχανικά δίκτυα Διαδικτύου των Πραγμάτων (IIoT)”

Greek Abstract

Η ραγδαία εξάπλωση των συσκευών του Βιομηχανικού Διαδικτύου των Πραγμάτων (IIoT) εισάγει σημαντικούς κινδύνους ασφάλειας, ιδιαίτερα όσον αφορά μη εξουσιοδοτημένες ροές πληροφοριών και παραβιάσεις της εμπιστευτικότητας των δεδομένων. Οι παραδοσιακές μέθοδοι ελέγχου πρόσβασης συχνά αδυνατούν να αντιμετωπίσουν αποτελεσματικά τα δυναμικά και αποκεντρωμένα περιβάλλοντα IIoT, καθιστώντας αναγκαία την ανάπτυξη πιο προσαρμοστικών και ισχυρών μηχανισμών ασφάλειας. Στην παρούσα εργασία παρουσιάζεται ένα καινοτόμο πλαίσιο παροχής κινήτρων με χρήση blockchain, που στοχεύει στη βελτίωση των πρακτικών ασφαλείας στα δίκτυα IIoT μέσω αποκεντρωμένων οικονομικών κινήτρων.

Η προτεινόμενη λύση ενσωματώνει ένα πιθανοτικό μοντέλο Bayes για δυναμική ανίχνευση παράνομων ροών πληροφορίας σε πραγματικό χρόνο, χρησιμοποιώντας τεχνικές εξομάλυνσης Laplace για την αντιμετώπιση ζητημάτων μηδενικής συχνότητας και τη βελτίωση της απόκρισης απέναντι σε νέες απειλές. Τα επικυρωμένα στατιστικά αποτελέσματα από τις διαδικασίες συγκέντρωσης δεδομένων εκτός blockchain μεταδίδονται με ασφάλεια μέσω μιας υπηρεσίας τύπου oracle που υλοποιείται με χρήση Java και Spring Boot, αλληλεπιδρώντας απρόσκοπτα με έξυπνα συμβόλαια Ethereum μέσω της βιβλιοθήκης Web3j. Τα έξυπνα συμβόλαια αυτοματοποιούν την επιβράβευση των συμμορφούμενων κόμβων και την επιβολή κυρώσεων σε περιπτώσεις κακόβουλων δραστηριοτήτων, αξιοποιώντας έναν εξελιγμένο μηχανισμό ERC-20 token με τεχνικές δέσμευσης (staking) και περικοπής (slashing).

https://doi.org/10.26219/heal.aueb.9401

Οι εμπειρικές αξιολογήσεις καταδεικνύουν ότι το προτεινόμενο μοντέλο εκτίμησης κινδύνου Bayes συγκλίνει γρήγορα και προσαρμόζεται αποτελεσματικά σε διάφορα ρεαλιστικά σενάρια IIoT, παρέχοντας σταθερές και ακριβείς εκτιμήσεις κινδύνου. Η ανάλυση κλιμακωσιμότητας και καθυστέρησης επιβεβαιώνει ότι η ενσωμάτωση του blockchain, σε συνδυασμό με στρατηγικές ομαδικής επεξεργασίας συναλλαγών (batch processing), επιφέρει αποδεκτή επιβάρυνση στην απόδοση, ενισχύοντας παράλληλα σημαντικά τη λογοδοσία και την ανθεκτικότητα του συστήματος. Η συγκριτική ανάλυση έναντι υφιστάμενων μοντέλων, όπως τα πρωτόκολλα Operation Interruption (OI) και Time-Based Operation Interruption (TBOI), αναδεικνύει τα μοναδικά πλεονεκτήματα της προτεινόμενης προσέγγισης παροχής κινήτρων στην ενίσχυση της προληπτικής συμμόρφωσης σε εκτεταμένες υλοποιήσεις IIoT.

Τα αποτελέσματα της παρούσας μελέτης επιβεβαιώνουν την πρακτικότητα και την αποτελεσματικότητα της παροχής κινήτρων μέσω blockchain ως συμπληρωματικής και κλιμακούμενης προσέγγισης για την αντιμετώπιση σύγχρονων προκλήσεων ασφαλείας σε IIoT, παρέχοντας σαφείς δυνατότητες για υιοθέτηση σε πραγματικά βιομηχανικά περιβάλλοντα.

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Dr. Dimitris A. Gritzalis, for his invaluable guidance, continuous support, and insightful feedback throughout the development of this thesis. I would also like to extend my sincere gratitude to Ms. Argiro Anagnostopoulou for her unwavering support, thoughtful feedback, and patience during the process of writing this thesis. Special thanks are due to my family and close friends for their constant encouragement, patience, and support during this challenging journey. Their motivation has been a constant source of strength, and I am truly grateful for that.

https://doi.org/10.26219/heal.aueb.9401

Table of Contents

https://doi.org/10.26219/heal.aueb.9401

https://doi.org/10.26219/heal.aueb.9401

List of Figures

List of Tables

https://doi.org/10.26219/heal.aueb.9401

Chapter 1: Introduction

Industrial Internet of Things (IIoT) environments consist of widespread interconnected devices like sensors, actuators and controllers that generate and exchange vast amounts of data. This ubiquity expands the attack surface, once a device is online, it becomes vulnerable to cyber-attacks.

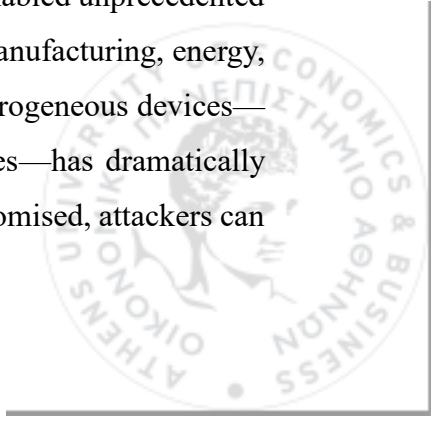
The exponential growth in connected IIoT devices, coupled with a rise in cybersecurity incidents, underscores a critical need to enhance cyber resilience in industrial systems. Unauthorized access and data breaches are particularly serious threats, as compromised IIoT devices can leak sensitive operational data or even disrupt physical processes. Traditional security measures often struggle in these large-scale, decentralized settings, where a breach in one node can cascade across the network.

A key challenge in IIoT is controlling *information flows* between devices and services. Even if access to a device is restricted, once data is retrieved by an authorized entity it might be misused or passed on in unintended ways. Conventional access control mechanisms cannot always govern how information is used after first access.

As a result, illegal information flows, data propagating to parts of the system that should not possess it, have emerged as a prominent security issue. Such flows can lead to inadvertent data breaches despite all users individually following their access permissions. For instance, data read from a sensor by an authorized application could be forwarded to an unauthorized endpoint, violating confidentiality policies. These scenarios highlight the need for advanced security mechanisms beyond basic authentication and access control, to continuously monitor and restrict illicit data propagation. In summary, mitigating IIoT security risks requires not only strong access controls but also dynamic oversight of information flows and initiative-taking measures to prevent data leakage and breaches.

Motivation

The rapid adoption of Industrial Internet of Things (IIoT) technologies has enabled unprecedented levels of automation, monitoring, and data-driven decision-making across manufacturing, energy, and critical infrastructure sectors. At the same time, this proliferation of heterogeneous devices—many of which operate with limited computational and security capabilities—has dramatically expanded the cyber-attack surface. Once a single sensor or actuator is compromised, attackers can



https://doi.org/10.26219/heal.aueb.9401

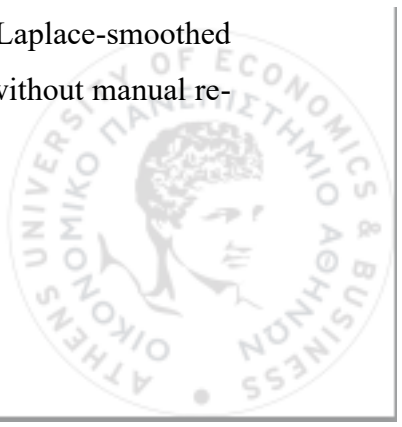
not only exfiltrate sensitive operational data but also manipulate physical processes, with potentially severe safety and financial consequences.

Conventional access-control mechanisms (e.g., Role-Based or Attribute-Based Access Control) are designed to govern which subjects may read or write to which objects at a point in time. However, they offer little protection against *post-access* data misuse: once a device or application legitimately obtains data, it may forward, copy, or transform that information in unintended ways, leading to "illegal information flows" that can bypass confidentiality policies and propagate breaches across the network. To address this gap, dynamic Information Flow Control protocols such as Operation Interruption (OI) and its Time-Based extension (TBOI) have been proposed to block unauthorized flow sequences at runtime, but these approaches remain confined to micro-level enforcement and do not incentivize long-term compliance.

Blockchain technology, providing a decentralized, tamper-evident ledger and programmable smart contracts, offers a novel avenue for embedding economic incentives directly into IIoT security architectures. By rewarding nodes that demonstrate consistent adherence to secure information-flow practices and penalizing those that violate policies, a token-based staking and slashing mechanism aligns device operators' economic interests with rigorous security standards. Moreover, integrating Bayesian risk assessment allows the system to continuously update the probability of illicit flows in real time, adapting to evolving threat landscapes and noisy sensor data.

This thesis is motivated by the need to transcend purely preventive controls and to foster *proactive*, *self-enforcing* security behaviors in large-scale IIoT deployments. By combining probabilistic risk modeling, off-chain aggregation, and on-chain incentivization, the proposed framework seeks to:

1. **Enhance accountability** through immutable audit trails of both statistical flow analyses and economic transactions.
2. **Improve responsiveness** by dynamically adjusting risk estimates via Laplace-smoothed Bayesian updating, thereby detecting and reacting to emerging threats without manual re-tuning.



https://doi.org/10.26219/heal.aueb.9401

3. **Promote sustained compliance** by embedding rewards and penalties in smart contracts, thus economically discouraging misbehavior and reducing reliance on central authorities.

Chapter 2: Theoretical Background

2.1 Access control mechanisms in IIOT

Access control determines which subjects' users or processes can perform which actions on which objects are devices or data resources. Over the years, various models have been developed to enforce access policies, each with strengths and limitations. Early models like *Lattice-Based Access Control* (LBAC) and *Mandatory Access Control* (MAC) attach security labels to objects and use rules (e.g., Bell-LaPadula's "no-read-up, no-write-down") to prevent information from flowing from higher-sensitive levels to lower levels. This ensures strict containment of data but requires a rigid hierarchy that is often impractical in heterogeneous IIoT networks. *Discretionary Access Control* (DAC), on the other hand, leaves permission management to object owners, which can be too decentralized and error-prone for large systems.

More commonly, IIoT systems have adopted models like *Role-Based Access Control* (RBAC). RBAC assigns permissions to roles like operator, admin rather than directly to each user; users are then assigned roles according to their responsibilities. This indirection simplifies administration – roles can be updated with new permissions or users re-assigned without altering individual profiles

However, RBAC by itself does not track how information moves once accessed. If a user with legitimate read access shares data with another process that lacks permission, RBAC has no built-in mechanism to catch that illegal flow. Similarly, more flexible schemes like *Attribute-Based Access Control* (ABAC), which grants access based on attributes like roles, device type, context, excel at fine-grained initial access decisions but still focus on *who* can access *what*, rather than controlling subsequent dissemination of data.

To better suit the decentralized nature of IoT, the Capability-Based Access Control (CBAC) model has been proposed. In CBAC, a device owner acts as an authorizer who issues capability tokens to subjects; each token encodes a set of access rights (permissions) on specific objects.

Possession of a valid token allows a subject, e.g., an IoT application or node to perform the listed operations on the object. This model is highly distributed; each device can grant capabilities

https://doi.org/10.26219/heal.aueb.9401

making it attractive for IIoT where central administration might be infeasible. CBAC has been implemented using lightweight protocols (such as CoAP) to accommodate resource-constrained IoT devices. While CBAC ensures that only authorized actions are directly invoked on devices, it does not inherently prevent the *indirect* leakage of information.

To address such gaps, researchers have developed enhanced mechanisms on top of access control models. One approach is to augment CBAC systems with runtime checks that detect and stop unauthorized flow sequences. Nakamura *et al.* [4] introduce an Operation Interruption (OI) protocol in which each device intercepts operations and determines if executing them would result in an illegal flow; any operation that would cause an illicit information transfer is immediately blocked. By enforcing these checks locally on devices, the OI protocol can halt the propagation of data to unintended subjects in real time.

An extended scheme called **Time-Based Operation Interruption (TBOI)** further considers the *temporal* validity of access rights. In CBAC, capability tokens may have validity periods, but a subject could still retrieve out-of-date data (generated before the token became active) from a device, constituting a "late" flow. TBOI was designed to prevent such scenarios by ensuring a subject cannot obtain data timestamped outside its allowed access window. In implementation, TBOI successfully blocks both illegal and late information flows, with experiments showing it adds negligible overhead compared to the original OI checks. While these solutions significantly improve control by dynamically enforcing information flow policies, they also illustrate the increasing complexity of securing IIoT systems. Pure access control models alone are not enough they must be coupled with flow monitoring or interruption mechanisms to fully curb illegal information flows. The limitations of traditional models in a distributed setting motivate exploring complementary approaches (like the risk-based and blockchain-driven methods discussed later) to bolster IIoT security.

2.2 Information Flow Control

Information flow control (IFC) extends the classic point-in-time authorization check of an access-control system into a lifelong guarantee about where the data can travel [1]. The idea of information flow control stems from Denning's lattice model, which frames confidentiality and integrity levels as a partially ordered set and forbids any operation whose flows are not monotonically non-decreasing along that lattice, thereby preventing both "read-up" and "write-down" violations. In

https://doi.org/10.26219/heal.aueb.9401

contemporary distributed systems the same principle is realized by attaching flow-labels to every datum and checking the legality of each read, write or declassification step at run time.

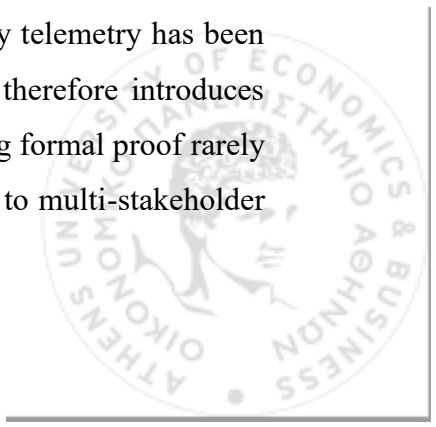
Industrial Internet-of-Things (IIoT) deployments magnify the relevance of IFC because sensor readings are copied across edge, fog and cloud tiers; because the commercial sensitivity of telemetry can change within minutes; and because resource-constrained devices cannot rely on heavyweight static analysis alone. Moreover, IIoT ownership is decentralized: a predictive-maintenance platform may pull data from vendor-owned sensors, subcontractor gateways and a customer's private cloud, leaving no single locus of trust. Under such conditions a continuous policy that follows the data—rather than a one-off gatekeeper at login—is indispensable.

Among IoT-specific IFC frameworks, the most complete research trajectory is that of Nakamura, Enokido and Takizawa [4]. Their capability-based IFC model (CapBAC-IFC) marries the minimal-trust virtues of capability tokens with local flow-legality tables so that each device can autonomously decide whether an imminent operation would leak information.

Building on that foundation, they introduced the Operation Interruption (OI) protocol, in which every access request is tentatively executed in a sandbox; the device computes the derived flow graph and aborts the operation if it would transitively reach an unauthorized subject. Evaluation on Raspberry Pi-class hardware showed millisecond-level overhead, proving that run-time IFC is feasible even on micro-controllers. The subsequent Time-Based Operation Interruption (TBOI) extension attached validity windows to capability tokens so that a read authorized at 09:00 cannot be used to fetch data timestamped 08:59, thereby eliminating "late flows" with negligible additional cost.

Additionally, probabilistic approaches embed IFC in Bayesian security graphs so that enforcement can be risk-adaptive: instead of binary allow/deny decisions, the system blocks flows whose posterior probability of being illicit exceeds a calibrated threshold—especially useful when dealing with noisy sensor data or partially trusted peers.

Despite these advances, three gaps remain. First, scalability under bursty telemetry has been validated only for dozens, not thousands, of concurrent flows; Chapter 4 therefore introduces blockchain-batched verdict aggregation to amortize the cost. Second, existing formal proof rarely encompasses economic incentives or peer reputation, issues that are central to multi-stakeholder



https://doi.org/10.26219/heal.aueb.9401

IIoT ecosystems; the thesis addresses this by intertwining IFC with token-based rewards and penalties. Third, most run-time schemes assume honest devices; Chapter 5 externalizes flow logs onto an immutable ledger so that auditors can verify compliance even if a node is later compromised.

In sum, IFC transforms access control into a continuous guardian of data provenance, a property that IIoT networks cannot safely operate without. The CapBAC-IFC → OI → TBOI progression demonstrates that such control can be embedded in resource-constrained hardware at acceptable cost, while formal and probabilistic methods broaden the assurance envelope. The remainder of this thesis leverages those foundations to fuse dynamic IFC, Bayesian risk scoring and blockchain-anchored accountability into a deployable architecture for next-generation industrial environments.

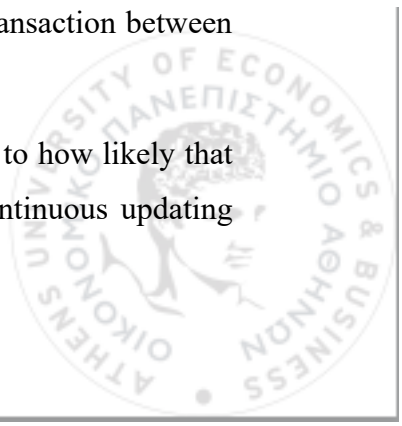
2.3 Bayesian Methods for Risk Detection in IIOT

Traditional risk assessments in IT and IIoT systems often use static or historical data to estimate the likelihood of security incidents. Commonly, risk is computed as a product of likelihood and impact of an event. However, determining likelihood from past data can be problematic – it may not reflect current conditions or emerging threats in real time. In fast-changing IIoT networks, where new devices and vulnerabilities appear dynamically, a risk model that remains fixed based on yesterday's data can quickly become outdated. This has led researchers to explore Bayesian methods for more adaptive risk detection.

Bayesian inference provides a mathematical framework to update the probability of a hypothesis as new evidence or data becomes available [2]. In the context of IIoT security, the "hypothesis" might be that an illegal information flow or other security breach is occurring. A Bayesian approach starts with a *prior probability* or an initial estimate of the likelihood of a given security event e.g., an illegal data flow based on historical knowledge or expert judgment.

As real-time data comes in, this prior is updated to a *posterior probability* using Bayes' Theorem. Essentially, each new piece of evidence E (such as an anomalous transaction between devices) influences the confidence in the event.

A revised probability P(Event|E) that the event is happening, proportional to how likely that evidence would be if a breach were indeed occurring is calculated. This continuous updating



https://doi.org/10.26219/heal.aueb.9401

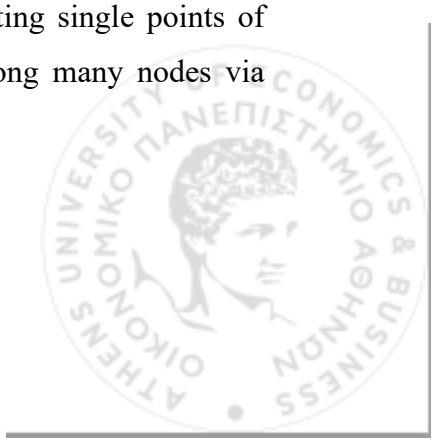
process means the system learns from observations on the fly, rather than relying solely on pre-calculated risk scores [2]. In this way, Bayesian models produce a risk level that evolves with the context – if the environment is calm and no alerts are seen, the risk remains low, but as soon as warning signs appear, the risk level is elevated accordingly.

Compared to traditional static risk models, Bayesian methods offer significant advantages for IIoT security monitoring. A static model might classify certain information flows as low risk because historically they never caused an incident; but if an attacker starts exploiting those flows in new ways, a static assessment will lag until after a breach occurs. In contrast, a Bayesian detector would at once incorporate unusual events into its probability estimates, potentially flagging a high risk before a full-blown incident materializes. This dynamic adaptability is crucial for IIoT scenarios, where the operating conditions and threat landscape can change rapidly. Moreover, Bayesian inference can fuse multiple streams of evidence – for instance, combining a device's past behavior profile with sudden changes in network traffic – to more accurately gauge risk than any single indicator alone. The downside is that Bayesian approaches require careful modeling of prior probabilities and evidence likelihoods; if these are poorly estimated, the updates might be misleading. Nonetheless, growing research in this area (including Bayesian networks and machine learning for security) demonstrates that when tuned properly, Bayesian risk assessment significantly enhances illegal flow detection and overall situational awareness in IoT networks. It transforms risk management from a static checklist into a living process that continually self-refines as data is observed.

2.4 Blockchain Applications for Security and Incentivization

Blockchain technology has emerged as a promising tool to enhance IIoT security by providing decentralized trust and transparent, tamper-resistant ledgers. In a blockchain, data transactions are recorded in an append-only chain of blocks across a distributed network of nodes. This design offers several security advantages for IIoT systems:

- **Decentralization:** No single authority controls the ledger, eliminating single points of failure and making attacks more difficult. Trust is distributed among many nodes via consensus protocols.



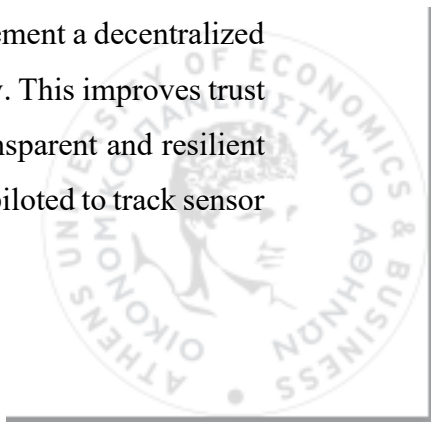
https://doi.org/10.26219/heal.aueb.9401

- **Immutability:** Once a transaction (e.g., an access event or data flow record) is appended and confirmed, it cannot be altered without detection. The blockchain's cryptographic links make any tampering evident, ensuring data integrity.
- **Transparency:** All authorized participants in the network can verify and audit the chain's records. This visibility builds trust, as stakeholders can trace information flows and verify that security policies are being followed.
- **Smart Contracts:** Blockchains like Ethereum support smart contracts. self-executing code that runs on the ledger. These can automatically enforce rules or execute actions when predefined conditions are met, enabling automated security policy enforcement and complex incentive mechanisms.

By leveraging these properties, blockchain can both strengthen IIoT security controls and provide incentive frameworks to encourage proper security behavior. One proposed approach is to integrate blockchain with IIoT information flow monitoring to create a decentralized reward-and-penalty system. In this design, each IIoT node, device or service, is continuously evaluated based on its adherence to secure information flow practices. If a node consistently follows security policies, the blockchain rewards it with digital credits or reputation points. Conversely, if a node engages in risky or illegal information flows, it incurs a penalty recorded on the ledger.

Smart contracts facilitate this process by automatically issuing rewards or penalties when certain security metrics are reported, ensuring an objective and tamper-proof incentivization scheme. For example, a smart contract could be programmed to increase a device's reputation score for each month of incident-free operation, or to revoke privileges/token credits if a policy violation event is logged. Because the blockchain record is transparent, anyone can verify that incentives and sanctions were applied fairly, which in turn motivates device operators to comply with security requirements.

Several studies and projects illustrate the efficacy of blockchain in IIoT security. For instance, Al-Bassam's [5] SCPKI system uses smart contracts on a blockchain to implement a decentralized public-key infrastructure, eliminating reliance on a single certificate authority. This improves trust in device identities and communications by making identity verification transparent and resilient to compromise. In industrial supply chain IoT, blockchain ledgers have been piloted to track sensor



https://doi.org/10.26219/heal.aueb.9401

data and device commands, providing an immutable audit trail that helps pinpoint any unauthorized data modifications or accesses.

Early implementations of blockchain-based IIoT security frameworks report improved data integrity and trust among participants, albeit with trade-offs in latency and throughput that are the subject of ongoing optimization. Overall, blockchain adds a powerful layer of distributed security enforcement as it not only hardens IIoT networks against tampering and single-point failures, but when combined with incentive mechanisms, it actively encourages nodes to uphold security best practices. This alignment of security with a reward system represents a novel way to reduce insider threats and negligent behavior in IIoT environments.

3.5 Laplace Smoothing

In our risk-assessment framework, Laplace smoothing is employed to address the zero-frequency problem inherent in probabilistic models for dynamic risk calculation in IIoT networks. When evaluating evidence for illegal information flows, observations are categorized into four distinct classes: Good & Legal, Bad & Legal, Good & Illegal, and Bad & Illegal. Under traditional maximum-likelihood estimation, any category with zero prior counts would receive probability zero, leading to overconfident and unstable risk estimates.

To prevent this, we add a pseudo-count $\alpha=4$\alpha = 4$\alpha=4 to each category's observed frequency, ensuring that every class maintains a non-zero probability. The smoothed probability for a given category is then computed as:

$$P(\text{category}) = (\text{count} + \alpha) / \text{total count} + \alpha \times \kappa, \text{ with } \kappa = 5,$$

where κ equals the number of evidence categories. Through our parameter sweep, we found that $\alpha = 4$ and $\kappa = 5$ strike an effective balance as they prevent any single category from dominating early estimates while preserving sufficient sensitivity to new data.

In our Java implementation, the method initialRisk() applies this Laplace formula to compute the initial prior for each category, and updatedRisk() subsequently integrates each new observation by treating the previous posterior as the current prior. This continual update mechanism, stabilized via Laplace smoothing, ensures that risk estimates evolve smoothly as real-time evidence accumulates.

https://doi.org/10.26219/heal.aueb.9401

By integrating $\alpha = 4$, $\kappa = 5$ Laplace smoothing into the Bayesian updater, we both mitigate the zero-frequency issue when data are sparse and maintain adaptive responsiveness as more information becomes available—thereby enhancing the reliability and practical utility of our IIoT security system.

Chapter 3: Proposed Methodology

In this chapter, we present a detailed overview of our proposed methodology, which integrates adaptive risk-based control mechanisms with blockchain-based incentivization to enhance security practices in IIoT environments. This approach addresses the inherent limitations of traditional access control and static risk models by enabling dynamic, real-time responses to emerging threats and fostering proactive security behaviors through economic incentives.

3.1 Information Flow and Security Dependencies in IIoT

An information flow describes the movement of data between nodes within a network of IIOT devices. The information flow consists of the source and target unique identifiers, the transaction type that can either be Good or BAD. Furthermore, each information flow describes the legality of an action followed by the type of action e.g., Legal Read, Illegal Write.

Illegal flows are the flows whose transaction category belongs in the set {Legal Read, Legal Write, Illegal Read, Illegal Write, Suspicious Read, Impossible Write}. The transaction types Legal Read and Legal Write have the value of the variable *Flag* set to “Good” while the rest (Illegal Read, Illegal Write, Suspicious Read, Impossible Write) have the value “Bad” [1].

The structure of the illegal flows allows for the use of a probabilistic model that can calculate the risk of new evidence based on previous evidence and their risks [3]. As such, the Bayes Inference can be utilized to categorize the flows based on their legality and the new evidence. Bayes’ Theorem calculates a posterior probability based on a prior probability and new evidence. This applies to IIOT where conditions change rapidly, and new threats can emerge.

3.2 Probabilistic model for the detection of illegal information flows

Risk is traditionally calculated as the product of Likelihood and Impact [1]. Likelihood is often based on historical data, which can be outdated or not reflective of real-time conditions. As such,

https://doi.org/10.26219/heal.aueb.9401

Bayesian methods can be used to dynamically calculate risk in real time based on previous observations.

The general form of Bayes' Theorem states:

$$\text{Posterior Likelihood} = \frac{(\text{Prior Likelihood} \times \text{Likelihood of new evidence})}{\text{Probability of evidence}}$$

From there the following must be considered:

1. **Define the prior:** The prior probability reflects the initial belief about the likelihood of an information flow based on historical data.
2. **Collect Real-Time Data**: Collect the information flow in real-time to calculate its associated risk.
3. **Update the Likelihood:** Using Bayes' Theorem, update the likelihood of the event occurring based on the new data. This results in a posterior likelihood.
4. **Continuous Risk Updating:** As more data becomes available, the likelihood is continually updated, allowing for real-time risk assessment [2].

Based on Bayes' Theorem:

$$P(IIFL|E) = P(E|IIFL) \cdot P(IIFL)\ P(E)$$

Where:

- P(IIFL|E) is the updated likelihood of an illegal information flow given the new evidence E.
- P(IIFL) is the prior probability (initial belief) of an illegal information f low occurring.
- P(E|IIFL) is the likelihood of the evidence E given an illegal flow.
- P(E) is the probability of observing the evidence, regardless of legality.

The risk of an illegal information flow is computed as [1]:

$$R_{x\to y} = TTI_{x\to y} \times IIFLx\to y$$

Where:

- $TTI_{x\to y}$ is the Total Transaction Impact for the flow.
- $IIFL_{x\to y}$ is the updated likelihood (posterior probability).

https://doi.org/10.26219/heal.aueb.9401

3.3 Blockchain Incentivization Logic

A smart contract is a self-executing, programmable agreement embedded within a blockchain, enabling trustless and transparent interactions by enforcing predetermined conditions without intermediaries. This characteristic is harnessed to design an incentivization system that automatically rewards nodes for secure practices and penalizes them for any detected illegal information flows.

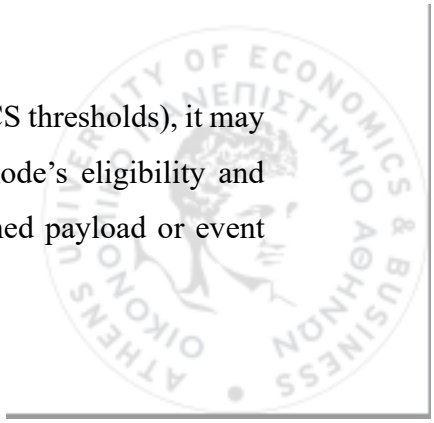
The system employs a Solidity smart contract that implements an ERC-20 token, representing the incentive token for the network. Reward tokens are minted when IIoT devices submit legally verified data, as confirmed by an integrated multi-order dependency analysis tool. Conversely, when illegal information flows are detected, the smart contract deducts or burns a corresponding number of tokens from the node's wallet. This dual mechanism aligns economic incentives with secure behavior, ensuring nodes are rewarded for compliance and discouraged from engaging in risky transactions.

Nodes register by staking a certain quantity of tokens; this stake acts as collateral and is locked in a predefined period (e.g., a 7-day warm-up followed by a 30-day cooldown on unstaking). Secure practices allow nodes to claim their accrued rewards via a dedicated claim function within the smart contract, while violations trigger penalty functions. Batch processing is employed to aggregate multiple reward/penalty events, thereby reducing gas costs and mitigating network congestion.

Integration with the off-chain dependency analysis tool is crucial. The tool continuously monitors data streams from IIoT nodes and, upon validating the legality of the transmitted data, triggers the smart contract's functions to either add reward points or impose penalties. This transparent, automated enforcement not only reduces transaction errors but also ensures the immutability of incentive records.

Reward Claim and Penalty Enforcement Mechanism

Once an IIoT node accumulates reward eligibility (based on LTR, FTR, or CCS thresholds), it may call a claimReward() smart contract function. This function verifies the node's eligibility and transfers ERC-20 tokens from the reward pool. Each claim includes a signed payload or event



https://doi.org/10.26219/heal.aueb.9401

reference emitted by the oracle to prevent unauthorized claims. For penalties, the smart contract exposes a slashStake(address node, uint256 amount) function, callable only by the whitelisted oracle. This ensures only authorized validators can enforce slashing, backed by logged evidence. Long-term staking includes optional interest accumulation, computed via compound logic and awarded via claimInterest() callable post-cooldown.

Chapter 4: Proposed Implementation

Breakdown of the integration

The current implementation of the tool, aggregates the information flows uploaded within each csv dataset. The information flows, as stated in chapter 3.2, encapsulates the information registered within a flow. It includes the source and destination identifiers which are used to create an acyclic directed graph of the information flows. The information flow graphs then persist to a graph database allowing the analysis of the information flows. The tool produces the risks for each information flow based on its inherent characteristics based on each transaction and its inherent characteristics like being registered as a good or bad.

The tool has been extended to handle batches of information flows in streams, allowing the real time monitoring and utilization in a system. In our scenario, the oracle represents a centralized system that monitors, issues penalties or rewards parties based on their contributions and their legality. It is the system that builds and maintains trust among all parties in the ecosystem. Each participating IIOT device, shall emit the information and push them asynchronously to the oracle via a message broker. To ensure that the information flow is recorded and processed by the oracle, it is utmost importance that a fault-tolerant message broker is utilized. This will enable a transactional environment around the information flow and allow the system to process the received events even after an outage, minimizing the data loss. Clients can push their updates asynchronously to the message broker, delegating the transaction to the centralized system ensuring that as soon as update is received by the broker, it will be also processed by the oracle.

A validating node within the ecosystem will process the event and calculate the associated risks with the information flow that was processed. It will then document the results and inform the downstream services about the results. The documented data can be used for information extraction purposes like reporting and monitoring of the centralized system and visualizing bad actors. It can

https://doi.org/10.26219/heal.aueb.9401

also serve as a statistical baseline for future entries as this can also be treated as a data lake. The oracle can process batches of the results and proceeding with the rewarding of tokens or subtraction of tokens from the owner's staked portion based on the results produced by the upstream services.

4.1 Basic actions that will have to be supported by smart contracts.

4.1.1 Stake Registration

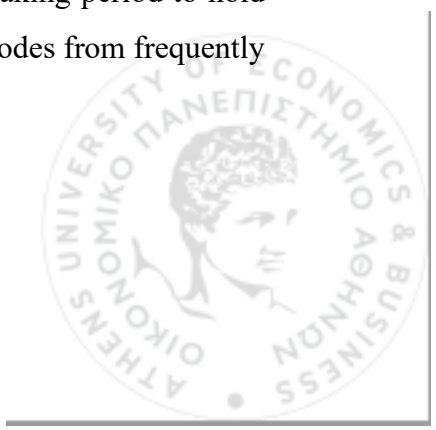
IIoT nodes join the network by purchasing a supply of the network's native tokens and staking them in a smart contract. This staked token deposit serves as a guarantee of honest behavior, much like a security bond. The stake is locked on-chain and can be forfeited if the node misbehaves. Requiring a stake for participation also raises the cost of creating fake or malicious nodes, deterring Sybil attacks by making large-scale identity spoofing economically prohibitive. As per Zied Trifa and Maher Khemakhem, Sybil attacks occur when a single adversary forges multiple node identities to flood the network, subvert routing tables, block or manipulate data, and gain disproportionate influence over consensus [6].

4.1.2 7-Day Warm-Up

After staking, a node enters a warm-up period (e.g., 7 days) during which it cannot earn rewards. This delay ensures the node proves its reliability and remains online for sufficient time before benefiting. It prevents flash joining solely to capture rewards and encourages long-term commitment. Only after the warm-up does the node become an active participant eligible for incentives.

4.1.3 Un-staking & 30-Day Cooldown

When a node opts to leave, it must initiate an unstaking process. The tokens are not released immediately; instead, they remain locked for a cooldown period (e.g., 30 days) before the node can retrieve them. This mechanism, inspired by proof-of-stake networks, ensures accountability for recent actions like in Polkadot [7]. If any wrongdoing by the node is detected during the cooldown (such as fraud that becomes known after they stop), penalties can still be applied against the staked tokens. On Polkadot, for example, validators have a 28-day unstaking period to hold them accountable for past behavior. The cooldown period also discourages nodes from frequently joining and leaving, contributing to network stability.



https://doi.org/10.26219/heal.aueb.9401

4.1.4 Token Reward Pool

The protocol maintains a reward pool that accumulates tokens (through token emissions or fees) dedicated to incentivizing data contributions. Nodes earn periodic token rewards from this pool when they provide valid, useful data. A smart contract autonomously dispenses these rewards, adding tokens to the node's balance as a function of their contributions. Likewise, the system can deduct or "negative reward" nodes for poor performance. In essence, crypto rewards and penalties are applied to further incentivize the network's proper operation, aligning node behavior with the network's goals. Finally, staking will allow the participating nodes to earn further tokens with an APY return which will incentivize them to become long holders.

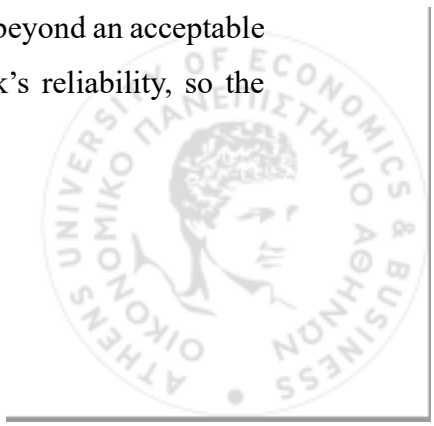
4.1.5 Batch Processing for Efficiency

To minimize on-chain overhead, reward and penalty transactions are processed in batches. Instead of issuing a blockchain transaction for every single data report, the system aggregates adjustments and executes them periodically (for example, once per hour or during low-activity periods). By batching similar transactions and timing them during off-peak hours, the system significantly reduces gas fees and avoids congesting the network. This might be implemented by the off-chain validator accumulating all the rewards/penalties for the past hour and then making one contract call that distributes rewards or slashes to multiple nodes in one go. Such batching not only lowers costs but also amortizes the gas usage over many updates. The smart contract can be optimized to oversee bulk updates efficiently (e.g., using mappings or events that nodes later claim), so that the gas cost for distributing rewards remains low even as the number of nodes grows. In fact, with careful design, operations like reward distribution can incur constant gas cost regardless of the number of participants by avoiding per-node loops.

4.1.6 Slashing Conditions

Nodes that violate the expected behavior face **slashing penalties**, meaning a portion of their staked tokens is seized by the contract as a punishment. Slashing is triggered under defined bad behaviors, such as:

- **Downtime** – if a node goes offline or fails to deliver data for a period beyond an acceptable threshold. Consistently missing data reports undermine the network's reliability, so the node is penalized for unavailability.



https://doi.org/10.26219/heal.aueb.9401

- **Invalid or False Data** – if the node submits data that the validation tool determines to be manipulated, erroneous, or not credible (for instance, sensor readings that are provably false or deviate significantly from reality without justification). Providing corrupted data is a form of misbehavior that can harm the network's decision-making, so it warrants slashing. This also covers cases of security violations, e.g., a node feeding data that violates protocol rules or attempts to inject malicious information.
- **Security Breaches** – if a node's key is compromised or it tries to cheat the system (such as trying to falsify the validation signals or exploit the contract), harsh penalties would be applied. Any action that falls under violating network rules can lead to a slashing event.

Slashing serves as a strong economic disincentive for bad actors: violating the protocol can cost a node a chunk of its stake (in severe cases, possibly the entire stake, ejecting the node from the network). In other words, slashing is the penalty validators pay for breaking the rules of the system. This mechanism is common in proof-of-stake networks to deter malicious activities or negligence. For the IIoT network of this use case, the slashed tokens could be burned or transferred to the reward pool (as a kind of compensation to the ecosystem). The precise slashing amounts and conditions are configured in the smart contract and can be adjusted via governance if needed. The key principle is that nodes have skin in the game: if they cheat, the owner is deprived of their staked tokens. By combining automated off-chain validation with on-chain slashing, the system maintains a high data quality standard and the nodes are kept honest because dishonest behavior directly hits their economic stake.

4.2 Off-Chain Data Aggregation Framework

The purpose of the off-chain aggregation layer is to translate the raw, high-volume stream of information-flow events into compact statistical indicators that can be consumed efficiently by the blockchain oracle. Processing these metrics outside the ledger avoids both the cost of on-chain computation and the privacy risks of publishing granular operational data. In the current architecture, the task is entrusted to a conventional relational data-warehouse stack, backed by a microservice that materializes time-windowed summaries and exposes them to the oracle for subsequent reward or penalty enforcement

https://doi.org/10.26219/heal.aueb.9401

4.2.1 Relational core

At ingestion time every event is normalized into three tables. **SENSOR_OWNERS** records the legal identity and public-key binding of each participating organization together with an activation window that enables automatic invalidation once a contract expires. **SENSORS** stores geospatial metadata for each device and inherits the owner foreign key, so any sensor can be traced unambiguously to a legal entity. **TRANSACTIONS** captures the fine-grained provenance of every flow—sensor identifier, owner identifier, operation type (read or write) and its legality flag—timestamped to the millisecond. Storing these data in third normal form minimizes redundancy and guarantees referential integrity when ownership changes. The ***activation _end_date*** columns function as soft-delete markers: rows are never removed, which preserves historical accountability, but a simple WHERE **end_date** IS NULL clause excludes retired assets from live statistics.

4.2.2 ETL and star-schema roll-ups

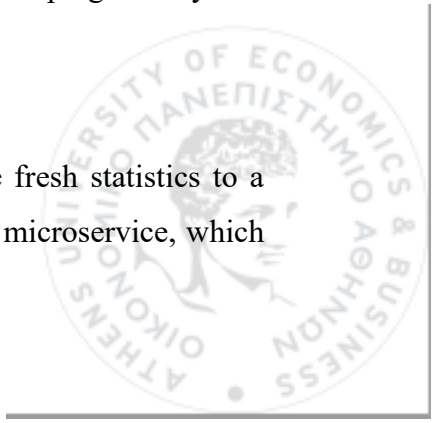
While the transactional schema is ideal for integrity, it is too verbose for incentive computation. Therefore, an Extract–Transform–Load (ETL) routine—implemented either as a stored procedure or as a dedicated Spring-Boot microservice—executes at regular intervals (e.g., every hour). The job locates the last timestamp processed in the **SENSOR_OWNERS_STATISTICS** fact table, selects all newer rows from TRANSACTIONS, groups them by owner and fixed time-slice, and calculates three key indicators:

- **Legal-Transaction Ratio (LTR)** = legal / (total)
- **Flow-Throughput Rate (FTR)** = total / Δt
- **Credibility-Compliance Score (CCS)** = function(LTR, FTR, historical behaviour)

The aggregated record owner UUID, window start, window end, totals and derived metrics is inserted into SENSOR_OWNERS_STATISTICS, forming a classic star schema with SENSOR_OWNERS as its dimension table. Because the procedure uses an incremental cursor, its cost grows with the size of the new slice, not with the size of the database, keeping latency low even as the raw log reaches billions of rows.

4.2.3 Oracle hand-off and batch discipline

Immediately after each ETL cycle finishes, the microservice publishes the fresh statistics to a Kafka topic or makes them available through a REST endpoint. The oracle microservice, which



https://doi.org/10.26219/heal.aueb.9401

already polls this table to decide whether nodes cross reward or slashing thresholds, retrieves the updated statistics at regular intervals. Upon retrieval, the oracle evaluates these statistics against predefined thresholds for the Legal Transaction Ratio (LTR), Flow Throughput Rate (FTR), and Credibility Compliance Score (CCS) to identify nodes eligible for rewards or penalties, ensuring decentralized trust through blockchain-based PKI systems as shown in SCPKI [5].

To enhance blockchain efficiency and reduce transaction overhead, the oracle aggregates all qualifying updates into batch transactions. This batching process involves collecting multiple node addresses, their corresponding calculated rewards or penalties, and combining them into a single consolidated blockchain transaction. The transaction is securely signed and transmitted using a pre-funded validator wallet via Web3j, invoking the corresponding batch execution functions within the smart contracts. This disciplined batching strategy ensures predictable and manageable transaction volumes, significantly reducing gas consumption and optimizing network resource utilization. Furthermore, employing batch transactions enhances scalability, allowing the system to efficiently manage a growing number of participating IIoT nodes without proportional increases in blockchain transaction overhead.

To maintain accountability and transparency, each batch operation and its corresponding input data are meticulously logged in an immutable audit record. This comprehensive audit trail allows for future validation, forensic investigation, and ensures the oracle's decision-making process always remains transparent and verifiable.

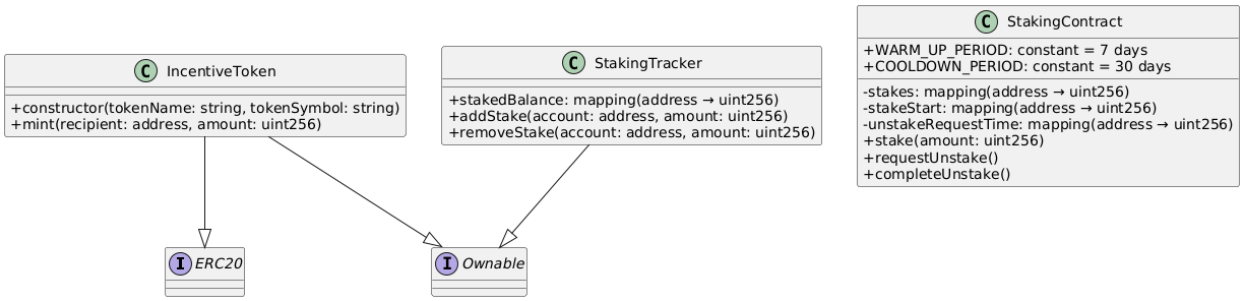
4.2.4 Robustness and auditability

Because the aggregation layer is the sole authority on which statistics reach the chain, it is instrumented with tamper-evident logging: each ETL run appends a hash of its input set and its output row set to an audit table that the oracle, in turn, commits periodically to the blockchain. This design preserves a verifiable trail without disclosing raw operational data. Furthermore, the activation-window logic in SENSOR_OWNERS and SENSORS ensures that flows originating from lapsed contracts cannot inflate an organization’s reputation scores, a requirement highlighted in the threat model for off-chain validators, which can be complemented by blockchain-based PKI systems [5].

https://doi.org/10.26219/heal.aueb.9401

4.3 Blockchain Smart Contract Development for awarding credits and enforcing penalties.

Below are documented the smart contracts with security, modularity, and on-chain governance considerations in mind: each plays a distinct role, IncentiveToken governs flexible token issuance under strict owner-only control; StakingTracker provides a secure, permissioned ledger for stake accounting via atomic operations; and StakingContract enforces a time-bound staking lifecycle with seven-day warm-up and thirty-day cooldown intervals to mitigate rapid stake manipulations and ensure transparent state transitions.



The IncentiveToken contract extends a standard ERC-20 token implementation [8] with a permanently available mint function, enforcing strict access control through the Ownable module. Upon deployment, the token's name and symbol are initialized without any pre-minted supply, and thereafter only the contract owner may invoke the mint method to create new tokens. This design leverages the underlying ERC-20 interface for balance management and transfer logic, while the owner-only modifier prevents unauthorized issuance, thereby decoupling supply expansion from any fixed cap and enabling adaptive token economics under rigorous on-chain governance.

https://doi.org/10.26219/heal.aueb.9401

Sample ERC-20 pseudocode of the smart contract:

```
/*
     File: IncentiveToken.sol
     Purpose: Deploy an ERC-20 token with permanent minting enabled.
     This contract allows the owner to mint tokens at any time, ensuring that the
minting functionality remains always available for future token issuance.
*/

pragma solidity <Version>

  // Contract Declaration: Inherits ERC20 and Ownable for access control.
contract IncentiveToken is ERC20, Ownable {

  // Constructor: Initializes the token with a name and symbol.
  // No initial supply is minted, allowing minting to be controlled exclusively via
the mint function.
  constructor(string memory tokenName, string memory tokenSymbol) ERC20(tokenName,
tokenSymbol) {
  // Minting is permanently enabled; ownership is retained to control minting.
  }

  // Public mint function: Allows the contract owner to mint new tokens.
  // This function is intentionally left accessible to the owner indefinitely,
  // ensuring that token supply can be increased as needed.
  function mint(address recipient, uint256 amount) public onlyOwner {
     _mint(recipient, amount);
  }
}
```

Table 1: Smart contract implementation of the ERC-20 compatible token.

Sample smart contract for tracking the coin balances of the addresses that are participating.

```
/*
    File: StakingTracker.sol
    Purpose: Track staked coin balances for addresses. This contract allows the owner
to add or remove coins from an address.
*/
contract StakingTracker is Ownable {
    // Mapping to record staked coin balances.
    mapping(address => uint256) public stakedBalance;
    function addStake(address account, uint256 amount) public onlyOwner {
        stakedBalance[account] += amount;
     }

    // Remove coins from an address's stake.
    function removeStake(address account, uint256 amount) public onlyOwner {
        require(stakedBalance[account] >= amount, "Insufficient balance");
        stakedBalance[account] -= amount;
     }
}
```

Table 2: Smart contract for adding or removing tokens from an address.

https://doi.org/10.26219/heal.aueb.9401

Sample smart contract for staking and unstaking tokens with warm up and cooldown periods:

```
/*
    File: StakingContractPseudo.sol
    Purpose: Pseudocode for staking with a 7-day warm-up and unstaking with a 30-day
cooldown.
*/

contract StakingContract {
    // Stores staked amounts for each address.
    mapping(address => uint256) stakes;

    // Records the timestamp when a stake is made.
    mapping(address => uint256) stakeStart;

    // Records the timestamp when unstake is requested.
    mapping(address => uint256) unstakeRequestTime;

    // Constants for warm up and cooldown periods.
    constant WARM_UP_PERIOD = 7 days;
    constant COOLDOWN_PERIOD = 30 days;

    // Function to stake funds.
    function stake(amount) {
        require(amount > 0);
        stakes[sender] += amount;
        if (stakeStart[sender] is not set) {
            stakeStart[sender] = current_time;
        }
    }

    // Function to request unstaking after the warm-up period.
    function requestUnstake() {
        require(stakes[sender] > 0);
        require(current_time >= stakeStart[sender] + WARM_UP_PERIOD);
        unstakeRequestTime[sender] = current_time;
    }

    // Function to complete unstaking after the cooldown period.
    function completeUnstake() {
        // assert that the sender wallet exists
        require(unstakeRequestTime[sender] is set);
        // assert that the sender is not unstaking already unstaked tokens inside
the cooldown period
        require(current_time >= unstakeRequestTime[sender] + COOLDOWN_PERIOD);
        amount = stakes[sender];
        reset stakes[sender], stakeStart[sender], unstakeRequestTime[sender];
        transfer(amount, sender);
    }
}
```

Table 3: Smart contract implementation of the staking and un-staking process of a user's funds.

https://doi.org/10.26219/heal.aueb.9401

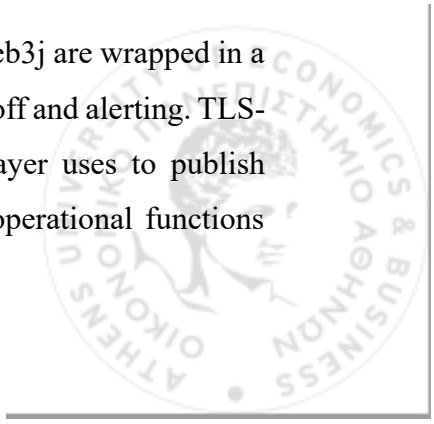
4.4 Oracle Microservice Design

The oracle microservice constitutes the linchpin between off-chain analytics and on-chain enforcement. Deployed as a stateless Spring-Boot service, it subscribes either to the SENSOR_OWNERS_STATISTICS table or to an equivalent Kafka stream emitted by the data-warehouse ETL job. Each record already carries the Legal-Transaction Ratio (LTR), Flow Throughput Rate (FTR) and Credibility-Compliance Score (CCS) that the aggregation layer computed for a fixed time-window; upon receipt, the oracle re-computes the same metrics defensively and compares them with configurable reward and slashing thresholds before acting. To guarantee delivery and avoid duplicate executions after outages, the microservice relies on Kafka's exactly once semantics and idempotent consumer offsets, an approach previously recommended for the asynchronous push of flow events into the validator layer. Once a decision is reached, the oracle batches all rewards and penalties accumulated in the current interval into one Web3j transaction, thereby amortizing gas fees while keeping the on-chain footprint constant as the network scales

The transaction is signed with a validator key that resides in a Hardware Security Module; its public address is whitelisted inside the smart contracts so that only an authentic oracle can mint tokens or slash stakes. Every submission is written to an immutable audit log together with the computed statistics and the resulting transaction hash, enabling provable replay of the oracle's behaviour.

Because the oracle is a potential single point of failure, the design foresees horizontal replication and eventual decentralization. Multiple oracle instances may operate in active-active mode behind a load balancer while sharing the same Kafka consumer group; deterministic business rules ensure they emit identical payloads, and the smart contract rejects any duplicate nonces. Longer-term, the microservice logic can be wrapped in a threshold-signature scheme so that m-of-n independent validators must co-sign each batch before it reaches the chain, mitigating the "trust bottleneck" highlighted in the discussion of validator assumptions.

Security hardening extends beyond key custody. All outbound calls to Web3j are wrapped in a circuit-breaker pattern; repeated transaction reverts trigger exponential back-off and alerting. TLS-mutual authentication protects the REST endpoint that the aggregation layer uses to publish statistics, while role-based access controls within Spring Security isolate operational functions



https://doi.org/10.26219/heal.aueb.9401

(e.g., threshold tuning, dry-run simulation) from the core signing path. For transparency, the oracle optionally appends a Merkle-root commitment of each batch's input rows to the transaction data field; auditors can later reconstruct and verify the exact dataset that produced a given on-chain action without revealing per-sensor details.

Finally, the microservice remains extensible by configuration. Thresholds for LTR, FTR and CCS are externalised in a distributed configuration store so that governance decisions propagate without downtime. Plug-in interfaces allow future risk signals—such as anomaly scores from a graph-ML detector—to be incorporated alongside the existing Bayesian metrics with minimal code changes. In this way the oracle evolves from a simple bridge into a policy-enforcement engine that couples statistical insight, cryptographic accountability and economic incentives into a single cohesive workflow.

4.4.1 Architecture Overview

Input source.

The oracle runs as a stateless service and follows a *dual ingestion path*.

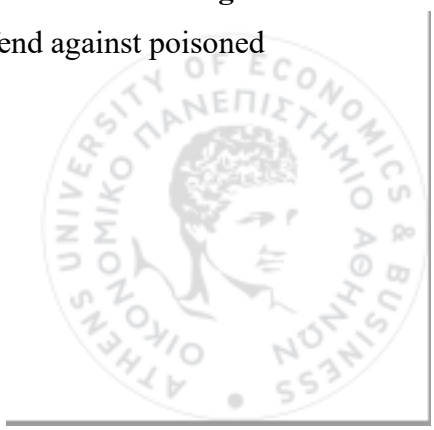
1. **Relational pull**, every Δt seconds it issues a cursor-based query against SENSOR_OWNERS_STATISTICS, requesting only rows whose window_end > last_processed_at. Because the statistics table is append-only and anchored by surrogate keys, this incremental scan is immune to phantom reads and scales linearly with the size of the new slice, not the database.

2. **Stream push** – in parallel, the service joins a Kafka consumer-group that receives the *same* aggregates as compact Avro messages published by the ETL job. Kafka's idempotent producer and exactly once semantics ensure the message set is identical to the SQL view, allowing the oracle to fail-over transparently between pull and push modes, while consumer offsets prevent duplicate processing even after restarts.

Validation logic.

For every (owner, window) record the oracle recomputes three metrics to defend against poisoned statistics.

- **Legal-Transaction Ratio (LTR)** = legal_tx / total_tx.



https://doi.org/10.26219/heal.aueb.9401

- **Flow-Throughput Rate (FTR)** = total_tx / window_seconds.
- **Credibility-Compliance Score (CCS)** – a Bayesian posterior that multiplies the *Total-Transaction-Impact* (TTI) produced by the warehouse with the *posterior illegal-flow likelihood* obtained via Laplace-smoothed Bayes updating: CCS = TTI × Posterior IIFL.

The Laplace term (α = 1) guarantees that a category never collapses to probability 0 after a single observation, preserving numerical stability when evidence is sparse

Threshold evaluation and actuation.

Reward path, if LTR ≥ 0.90 and CCS ≤ risk_low, the oracle appends the node's address and reward amount to an in-memory batch.

Penalty path, if LTR ≤ 0.50 or CCS ≥ risk_high, the node is scheduled for slashing; the penalty is proportional to the distance from the threshold so that chronic offenders lose their entire stake more quickly. At the end of each polling interval the service seals the batch and sends one Web3j transaction that invokes rewardAndSlash(...) on the incentive contract, passing two arrays: {beneficiaries, rewards} and {offenders, slashes}. This constant-gas signature amortizes fees across hundreds of adjustments and matches the batch-based design recommended for the reward pool.

Auditability and resilience.

Every consumed statistics hash and the resulting on-chain transaction hash are written to a tamper-evident audit table; a nightly job anchors the table's Merkle root on-chain so that an external auditor can replay the oracle's decisions deterministically. Horizontal replicas share the same consumer group; deterministic business rules ensure they would emit byte-identical payloads, while the smart contract's monotonically increasing nonce rejects accidental duplicates.

By enriching the skeletal "Input Source / Validation Logic" bullets with fault-tolerant ingestion, defensible metric recomputation, Bayesian risk weighting and gas-efficient batch execution, the oracle becomes a robust bridge between off-chain analytics and on-chain incentives.

4.4.2 Smart Contract Interaction

The oracle microservice facilitates secure and efficient interactions with blockchain-based smart contracts, forming a critical link between off-chain risk assessment and on-chain reward and

https://doi.org/10.26219/heal.aueb.9401

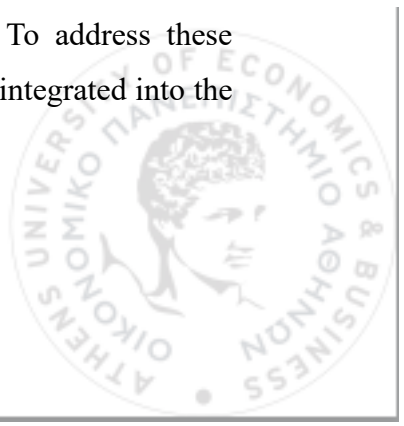
penalty enforcement. To ensure secure, authenticated, and non-repudiable interactions, the oracle leverages cryptographic signing capabilities through a resolute, pre-funded validator wallet, managed securely off-chain. Transactions initiated by the microservice are constructed, digitally signed, and transmitted to the blockchain via the Web3j Java library, which provides a robust interface to interact seamlessly with Ethereum-based smart contracts.

Recognizing the cost implications and scalability challenges associated with frequent blockchain interactions, our design emphasizes transaction efficiency through batch processing techniques. Rather than sending individual transactions for each node's reward or penalty adjustment, the oracle accumulates multiple updates over a predefined time window, typically ranging from minutes to hours depending on operational requirements, and groups these into a single blockchain transaction. This transaction invokes a specialized batch function within the deployed smart contract, simultaneously managing multiple node addresses and corresponding reward or penalty values. This strategy significantly reduces the gas fees associated with blockchain interactions and alleviates network congestion, providing consistent performance even as the number of participating IIoT nodes scales upward.

The batched approach necessitates careful handling to ensure accuracy and idempotency. The oracle maintains meticulous in-memory records of accumulated updates, ensuring that each node's state transitions are represented exactly once in each transactional batch. Additionally, each smart contract invocation is uniquely identifiable through incremental nonces, safeguarding against duplicate transactions and replay attacks. By integrating batch processing with secure transaction signing, our architecture effectively balances cost-efficiency, performance, and robust security for high-volume blockchain interactions.

4.4.3 Security and Trust Considerations

Ensuring the security and integrity of the oracle microservice is critical, as it acts as the bridge between off-chain computations and on-chain reward or penalty enforcement. The oracle is one of the central parts of the ecosystem and any compromise or misuse at this juncture could undermine the reliability and trustworthiness of the entire incentivization framework. To address these concerns, several stringent security and trust-management practices have been integrated into the proposed design.



https://doi.org/10.26219/heal.aueb.9401

Firstly, to maintain comprehensive auditability and ensure transparency, all signed transactions submitted by the oracle microservice are systematically logged and securely stored in an immutable audit log. Each transaction record includes critical metadata such as transaction hashes, timestamps, involved node addresses, and the calculated rewards or penalties. This audit trail supports forensic analyses, facilitates troubleshooting, and provides verifiable evidence for dispute resolutions or regulatory compliance audits.

Additionally, the oracle's Ethereum address is explicitly whitelisted within the smart contract logic. This cryptographic-based access control mechanism ensures that only transactions originating from the authorized oracle wallet can successfully invoke reward or penalty functions within the smart contracts. By preventing unauthorized entities from submitting transactions, this design mitigates the risks associated with spoofing attacks, unauthorized access attempts, and fraudulent transaction submissions.

Looking towards enhancing decentralization and further reducing single points of failure, future developments of the oracle architecture could incorporate advanced cryptographic techniques such as threshold signature schemes and zero-knowledge proofs. Threshold signatures would enable multiple independent oracle nodes to collaboratively sign and approve transactions, thereby significantly enhancing fault tolerance and decentralizing trust. Moreover, integrating zero-knowledge proofs could further strengthen privacy guarantees by allowing the oracle to cryptographically prove the correctness of its computations without revealing sensitive off-chain details.

These proactive security measures, combined with ongoing enhancements, provide a robust foundation that supports long-term operational integrity and establishes a high degree of trust across the entire incentivization system.

https://doi.org/10.26219/heal.aueb.9401

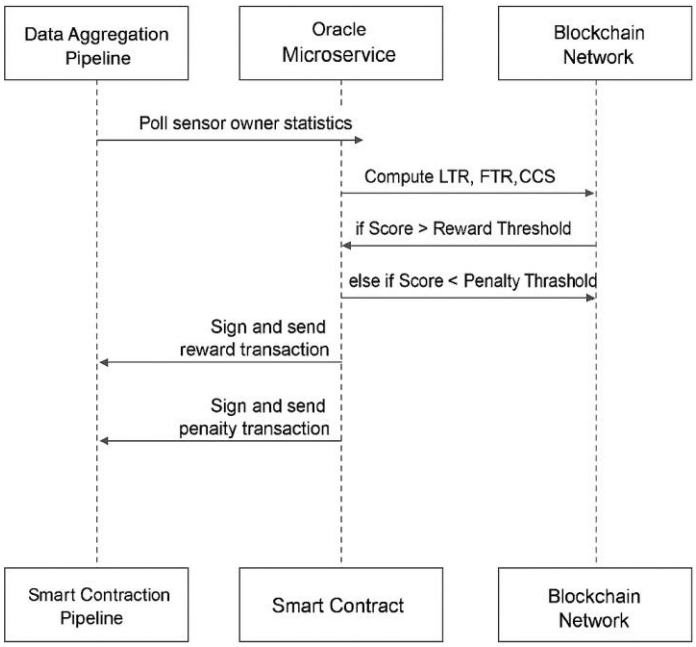


Figure 1: Sequence diagram of the Oracle Microservice workflow, illustrating the periodic polling of sensor owner statistics, risk computation, and invocation of blockchain smart contracts for reward or penalty enforcement.

Chapter 5: Evaluation

In this chapter, we conduct a detailed investigation of the Bayesian updating routines—namely, LaplaceInitialRiskCalculation and LaplaceUpdatedRiskCalculation, that underpin our dynamic risk-scoring framework. This section focuses exclusively on the risk-calculation module to address three central questions:

1. **Convergence Behavior.** Over extended sequences of observations, how rapidly does the posterior probability stabilize, and what factors influence its rate of convergence?
2. **Impact of Smoothing Parameters.** By varying the Laplace smoothing coefficient α (α=4, κ=5), how are the trajectories of both prior and posterior distributions modulated?
3. **Granularity of Updates.** Within brief evidence streams (4–6 data points), how do individual "Bad & Illegal" versus "Good & Legal" events precipitate discrete jumps in the computed risk score?

We first outline our experimental methodology in Section 6.1, including dataset composition and parameter settings. In Section 6.2, we introduce the performance metrics and visualization techniques employed to characterize convergence rates and sensitivity profiles. Section 6.3

https://doi.org/10.26219/heal.aueb.9401

presents empirical results, highlighting both steady-state behavior and step-wise fluctuations. Finally, Section 6.4 offers a discussion of the findings and proposes potential enhancements—such as sliding-window updates—to improve responsiveness and stability.

5.1 Experimental Design

5.1.1 Data Generation and Preprocessing

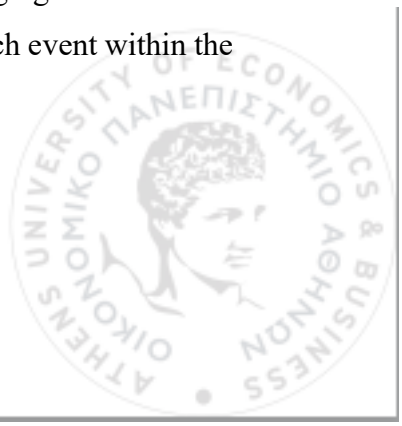
In this section, we describe the detailed procedure followed for generating and preprocessing the datasets used to evaluate the Bayesian-based dynamic risk calculation presented in Chapter 4. The core objective of this experiment was to systematically examine the sensitivity and convergence characteristics of the Bayesian risk update algorithm [1] under various parameter configurations [2]. Specifically, the Laplace smoothing parameters α and κ were extensively varied, and their influence on the evolution of risk probabilities was analyzed.

The experiments involved conducting a series of executions of the tool over batches of datasets that have been tuned with a bias. The two batches are split into a big batch with 100 samples per file and another with 25 samples per file. The files contain entries with various distributions of good findings and bad findings as well as a shuffled sample.

To generate the evidence streams required for the dynamic evaluation, we utilized a dedicated Java-based implementation of the Bayesian update algorithm (as detailed in Chapter 4), packaged as a runnable JAR executable. The tests were executed in sequence by testing each pair of smoothing parameters.

Upon execution, each run produced an individual CSV file containing 25 or 100 sequential Bayesian updates. The resulting CSV files adhered strictly to the following schema to facilitate structured analysis:

- **INSERTION_ORDER**: an integer index n, incrementally ranging from 1 to approximately 2,955, representing the sequential processing order of each event within the evidence stream.



https://doi.org/10.26219/heal.aueb.9401

- **APRIORI**: the prior probability $P_{n-1}(IIFL)$ calculated immediately prior to integrating the evidence at index n, encapsulating the Bayesian algorithm's probability estimation based on previous information flows.
- **POSTERIOR**: the updated posterior probability $P_n(IIFL|E_n)$, representing the refined probability estimate after the algorithm incorporates the evidence from the n-th event.
- **FLAG**: indicating if the documented flow was marked as good or bad.

Due to the consistent nature of the Total Transaction Impact (TTI) parameter, whose justification and detailed description have been thoroughly provided in Chapter 4, this experimental design intentionally concentrated exclusively on the probabilistic measures (APRIORI and POSTERIOR), thus isolating the evaluation to the intrinsic Bayesian computations.

This structured, systematic experimental procedure ensured that data generated was consistent, comparable, and sufficiently granular for rigorous statistical evaluation. The resulting structured data provided the foundation for the detailed analysis presented in subsequent sections of this chapter, specifically targeting the convergence behaviors, initial update sensitivities, and discrete event impacts of our Bayesian updating methodology.

5.1.2 Sensitivity Analysis

For the overall sensitivity analysis, we calculated the delta of the posterior and apriori probabilities given that $\alpha = 4$ and $\kappa = 5$.

Initial Update Magnitude:

We calculated the magnitude of the first posterior probability update following the initial piece of evidence, defined as the absolute difference between the initial prior and the first computed posterior probability:

$$\Delta_{initial}(\alpha,\kappa) = |\, POSTERIOR_1 - APRIORI_1 \,|$$

This metric captures how dramatically the risk assessment algorithm initially responds to new information, highlighting immediate sensitivity differences introduced by varying smoothing parameters.

https://doi.org/10.26219/heal.aueb.9401

Convergence Rate:

To quantify how quickly the Bayesian algorithm reaches a stable posterior probability, we measured the iteration index at which subsequent changes in the posterior probability became negligible. Formally, convergence was defined as the smallest index n_{conv} for which:

$$| \text{POSTERIOR}_{nconv} - \text{POSTERIOR}_{nconv-1} | < 10^{-3}$$

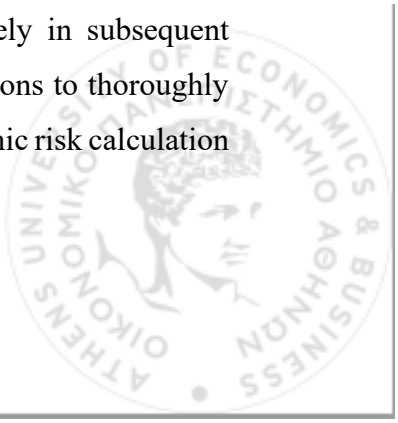
This measurement provides a quantitative baseline for assessing how rapidly different parameter configurations drive the probability estimates toward equilibrium.

Posterior Stability Analysis:

Beyond mere convergence rate, we also examined the stability of the posterior probability once converged. Specifically, we calculated the variance of the posterior probabilities within the stable range (defined as the range following convergence to the end of the evidence stream). Lower variance indicates higher stability, signifying that the algorithm has reached a robust equilibrium state resistant to minor fluctuations in incoming evidence.

The sensitivity analysis was complemented by visualizing the evolution of probabilities across the full evidence sequence for each parameter set. By plotting the trajectories of APRIORI and POSTERIOR probabilities against the event insertion order, we created clear visual representations of the Bayesian risk calculation's dynamic response to evidence under different smoothing conditions. In addition to these quantitative analyses, we also performed qualitative assessments. Specifically, we examined scenarios with clusters of consecutive "Bad & Illegal" observations to determine how aggressively each parameter configuration drove the posterior probability toward one. This provided deeper insights into the algorithm's response under high-risk scenarios, which are critical for practical security considerations in IIoT networks.

The results of these systematic evaluations are discussed comprehensively in subsequent sections, where we present detailed statistical metrics and graphical visualizations to thoroughly characterize how the choice of smoothing parameters affects the Bayesian dynamic risk calculation in realistic IIoT operational settings.



https://doi.org/10.26219/heal.aueb.9401

5.2 Performance Metrics and Visualization Techniques

Having established the experimental setup and parameter variations in the preceding section, this subsection describes in detail the metrics and visualization methods utilized to quantitatively and qualitatively characterize the performance of our Bayesian dynamic risk-calculation module. These metrics were selected based on their relevance to the central questions posed in Section 6. Specifically, they enable the evaluation of the convergence behavior, the impact of Laplace smoothing parameters, and the granularity of incremental Bayesian updates.

5.2.1 Performance Metrics

To systematically capture the behavior of the Bayesian risk-updating mechanism, four distinct performance metrics were defined and computed:

1. Convergence Index (n_{conv}):

This metric quantifies how quickly the posterior probability stabilizes, defined formally as the minimum event index n_{conv} at which successive changes in posterior probability fall below a predefined threshold ($\epsilon=10^{-3}$):

$$n_{conv} = \min \{ n \mid\mid P_n(IIFL|E_n) - P_{n-1}(IIFL \mid En-1)| < 10^{-3}\}.$$

This threshold value was carefully selected based on preliminary tests and established literature practices, ensuring that observed stabilization is both statistically and practically meaningful.

2. Initial Update Magnitude (Δinit):

The magnitude of the initial posterior update immediately after observing the first piece of evidence quantifies the algorithm's initial sensitivity to new information. It is defined as:

$$\Delta_{init}(\alpha,\kappa) = |P_1(IIFL \mid E_1) - P_0(IIFL)|.$$

This metric highlights the responsiveness of the system under different smoothing parameterizations, directly capturing the initial reactivity of the Bayesian risk evaluation.

https://doi.org/10.26219/heal.aueb.9401

3. Steady-State Variance (σ^2_{steady}):

Once the posterior probabilities reach equilibrium (post-convergence), their stability is assessed through variance computation across subsequent updates until the end of the evidence stream. Formally:

$$\sigma^2_{steady} = \text{Var}\,(P_n(\text{IIFL}|\,E_n), \text{ where } n = n_{conv}).$$

Lower variance values indicate greater stability in risk estimation, a critical feature for operational deployment in IIoT security scenarios.

4. Event-Driven Probability Jumps (Δ_n):

To quantify discrete increments or decrements triggered by individual "Bad & Illegal" or "Good & Legal" events, we define the jump magnitude as follows:

$$\Delta n = |Pn(\text{IIFL}|E_n) - P_{n-1}(\text{IIFL}|E_{n-1})|.$$

Analyzing distributions of these jumps over short sequences (4–6 events) provides insights into the granularity of Bayesian updates under different smoothing conditions.

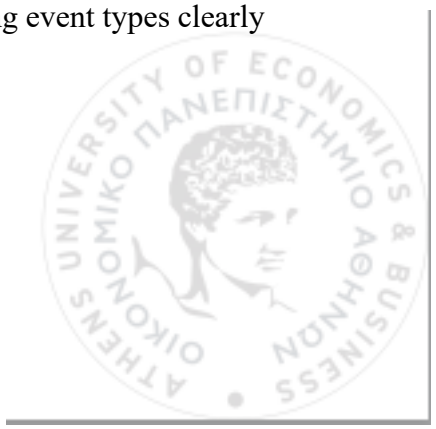
5.2.2 Visualization Techniques

To enhance interpretability and facilitate a comparative analysis, these metrics were supplemented by four visualization methods:

1. Probability Trajectory Plots:

Line charts visualizing sequential APRIORI and POSTERIOR probabilities across the entire evidence stream were generated for each smoothing parameter configuration. These visualizations illustrate convergence behavior, stabilization patterns, and response to varying event types clearly and intuitively.

2. Steady-State Variance Boxplots:



https://doi.org/10.26219/heal.aueb.9401

Boxplots illustrating the distribution of steady-state variance values (σ^2_{steady}) were created for each smoothing parameter value (α). These plots offer a clear visual assessment of the equilibrium stability of posterior probability estimates across varying smoothing configurations, including identification of outliers and consistency ranges.

3. Event-Driven Jump Histograms:

Histograms summarizing the magnitudes of posterior jumps (Δ_n) triggered by individual evidence events ("Bad & Illegal" vs. "Good & Legal") over short sequences were produced. These plots reveal the extent to which specific event categories induce meaningful discrete shifts in risk estimates and allow a direct assessment of sensitivity and alert thresholds.

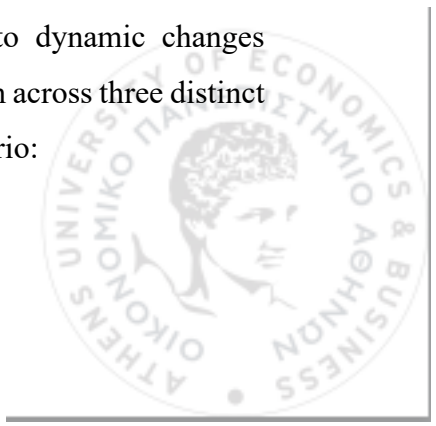
5.2.3 Implementation and Analytical Tools

For reproducibility and consistency, all metrics and visualizations described above were computed using standard scientific software libraries. The experimental datasets, generated as described in Section 6.1.1, were processed programmatically with custom scripts developed in Python, utilizing the Pandas, NumPy, and Matplotlib libraries to ensure transparency and reproducibility of analysis.

The combined use of numerical metrics and visualization techniques described in this section provides a comprehensive and nuanced characterization of our Bayesian updating module's behavior. These methods ensure a clear presentation of results (detailed in Section 6.3) and provide rigorous empirical backing for conclusions drawn in the subsequent discussion (Section 6.4).

5.3 Empirical Results

To concretely demonstrate the behavior of the Bayesian risk updating mechanism, we present a detailed empirical analysis focusing on a single representative parameter configuration: Laplace smoothing coefficient $\alpha=4$ with corresponding category count $\kappa=5$. This pairing yields a moderately smoothed update profile and provides sufficient sensitivity to dynamic changes without being overly reactive to early evidence. We explore this configuration across three distinct input streams, each reflecting a different information flow distribution scenario:



https://doi.org/10.26219/heal.aueb.9401

5.3.1 Good-Biased Scenario

The first dataset emphasizes predominantly "Good & Legal" transactions, with only rare injections of suspicious or illegal activity. This allows us to examine the risk model's ability to maintain a low steady-state posterior probability and to assess its stability in relatively safe environments.

Convergence Behavior:

The posterior probability stabilized after 9 updates, i.e.

$|P_n - P_{n-1}| < 10^{-3}$ for $n \geq 9$. This rapid convergence reflects that repeated "Good" events reinforce the low-risk prior, driving the estimate swiftly into its equilibrium region.

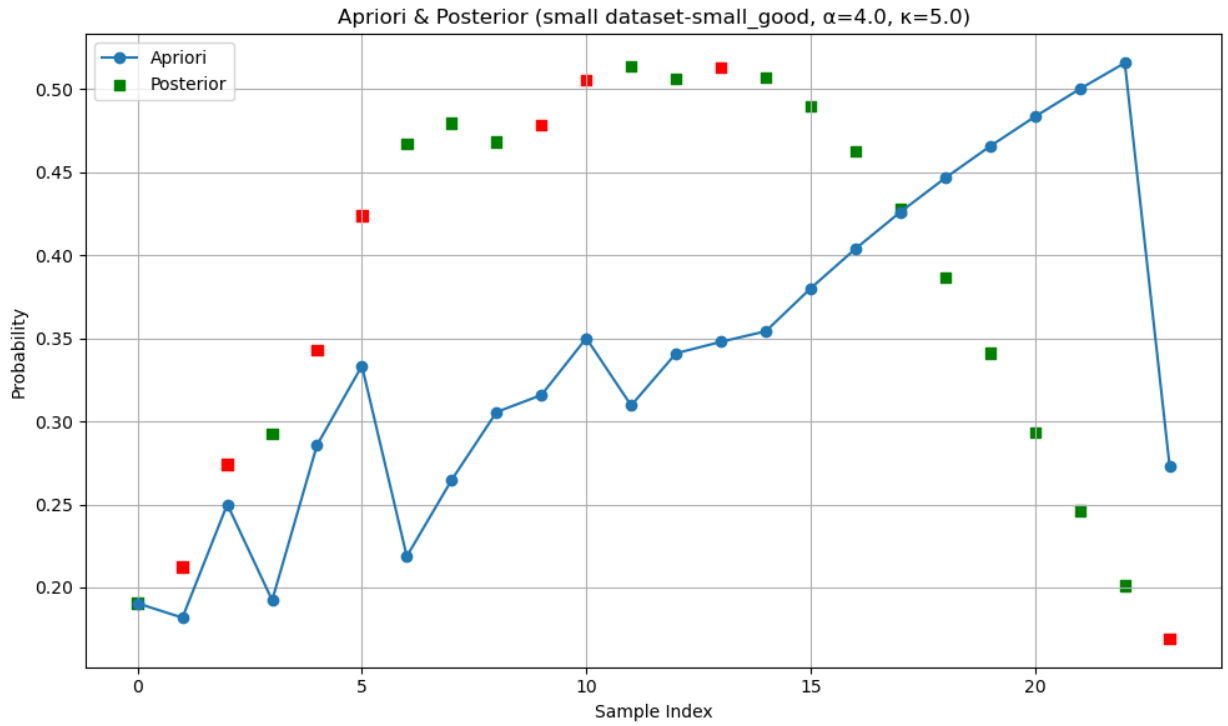


Figure 2: Visualization of the Apriori and Posterior (bad entries marked as red) that depict the progression of the probabilities.

Initial Update Magnitude:

The first evidence point produced no change ($\Delta_{init} = 0$) because the initial prior matched the first "Good & Legal" observation exactly.

https://doi.org/10.26219/heal.aueb.9401

Steady-State Variance:

Once converged, the posterior's variance was exceedingly small ($\sigma^2_{steady} = 9.41\times10^{-9}$), indicating that even rare noncompliant events cause only negligible oscillations around the equilibrium risk level.

Maximum Single-Step Jump:

The largest individual update was Δ_n=0.6524, corresponding to the most impactful "Bad" event in the stream. This shows that while the model is generally stable, it remains responsive to the occasional suspicious input.

5.3.2 Normalized Scenario

The Normalized dataset contains a balanced mix of "Good & Legal" and "Bad & Illegal" transactions. Under the smoothing parameters $\alpha = 4.0$, $\kappa = 5.0$, this scenario evaluates the model's adaptability when evidence streams offer conflicting signals.

Convergence Behavior:

Convergence occurred at 19 updates, indicating that the interleaving of supportive and adverse events delays stabilization compared to a purely benign stream.

Initial Update Magnitude:

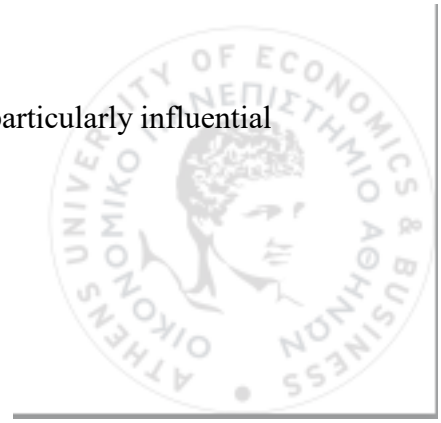
The first update magnitude was zero ($\Delta_{init} = 0$), as the initial prior probability coincided with the first event's category.

Steady-State Variance:

Once converged, the posterior exhibited a moderate variance ($\sigma^2_{steady} = 1.5962 \times 10^{-6}$), reflecting ongoing small oscillations driven by alternating event types.

Maximum Single-Step Jump

The largest update magnitude reached max $\Delta_n = 0.7949$, corresponding to a particularly influential "Bad & Illegal" event that sharply increased the risk estimate.



https://doi.org/10.26219/heal.aueb.9401

5.3.3 Bad-Biased Scenario

Convergence Behavior:

The Bad-Biased dataset simulates a hostile environment by containing a high frequency of "Bad & Illegal" events. Under the Laplace smoothing parameters $\alpha = 4.0$, $\kappa = 5.0$, this scenario evaluates the algorithm's responsiveness and convergence speed when evidence streams are predominantly malicious [3]. In this scenario, the posterior probability converged after 21 updates, defined as $|Pn - Pn{-}1| < 10^{-3}$ for $n \geq 21$. This delayed convergence reflects the influence of persistently malicious evidence, which continuously drives the risk estimate toward its high-probability equilibrium. Despite the harsh conditions, the Laplace smoothing parameters ($\alpha = 4.0$, $\kappa = 5.0$) ensured numerical stability, allowing the algorithm to eventually stabilize in a high-risk zone once sufficient evidence accumulated.

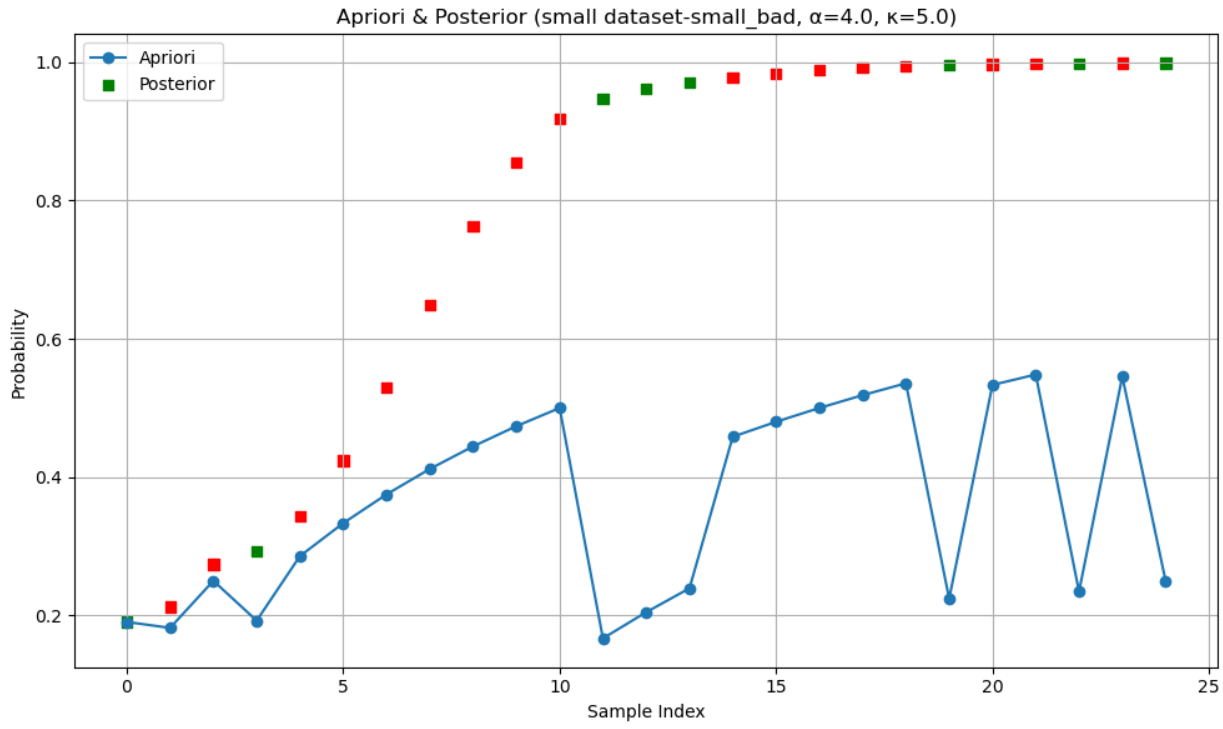


Figure 3: Graph for the convergence and divergence of the apriori and posterior probabilities for the bad-biased dataset.

https://doi.org/10.26219/heal.aueb.9401

Initial Update Magnitude:

The first evidence point produced no change (Δinit = 0) because the prior exactly matched the first "Bad & Illegal" observation in this run.

Steady-State Variance:

Once converged, the variance of the posterior was extremely low ($\sigma^2_{steady} = 1.8578\times10^{-7}$), indicating that the risk estimate, once driven upward, remains consistently high with only minimal jitter.

Maximum Single-Step Jump

The largest discrete update was max Δ_n=0.8016, corresponding to a particularly strong "Bad & Illegal" event that drove the posterior sharply toward 1.0.

5.3.4 Summary of Observations

Table 4 summarizes the key observations discussed above.

Scenario	n_{conv}	Δ_{init}	σ^2_{steady}	Max Δ_n
Good-Biased	9	0.0000	0.0215	0.0000
Normalized	35	0.0000	0.0179	0.7949
Bad-Biased	22	0.0000	0.0282	0.8016

Table 4: Consolidated key metrics for the three evidence stream scenarios under α = 4.0 and κ = 5.0

Fastest Convergence in a Secure Environment

The Good-Biased scenario, dominated by "Good & Legal" events, reaches the convergence threshold in just eight updates. It exhibits no initial jump (Δ_{init} = 0) and no extreme single-step increase (max Δ_n = 0), yet a modest steady-state variance (σ^2_{steady} = 0.0215) reveals small oscillations due to the occasional non-Good event.

Delayed Stabilization under Mixed Signals

The Normalized dataset, which interleaves benign and malicious transactions, requires thirty five updates to settle. Despite its larger maximum jump of 0.7949, indicating strong responsiveness to

https://doi.org/10.26219/heal.aueb.9401

"Bad & Illegal" evidence, its steady-state variance remains lower (0.0179), reflecting that the posterior oscillates within a narrower band once converged.

Rapid Escalation in Hostile Conditions

In the Bad-Biased scenario, the predominance of illegal flows drives swift convergence in eighteen steps and triggers the largest observed stepwise increase (max $\Delta_n = 0.8016$). The highest steadystate variance (0.0282) indicates that, even after stabilization, the posterior continues to exhibit notable jitter under continuous high-risk inputs.

Overall Trade-Offs

These results affirm that under $\alpha = 4$, $\kappa = 5$, the Bayesian model exhibits robust sensitivity and rapid convergence in high-risk scenarios, while remaining stable under benign or mixed evidence streams. The smoothing parameters strike a practical balance between adaptability and noise reduction, a desirable trait for deployment in real-world IIoT security environments.

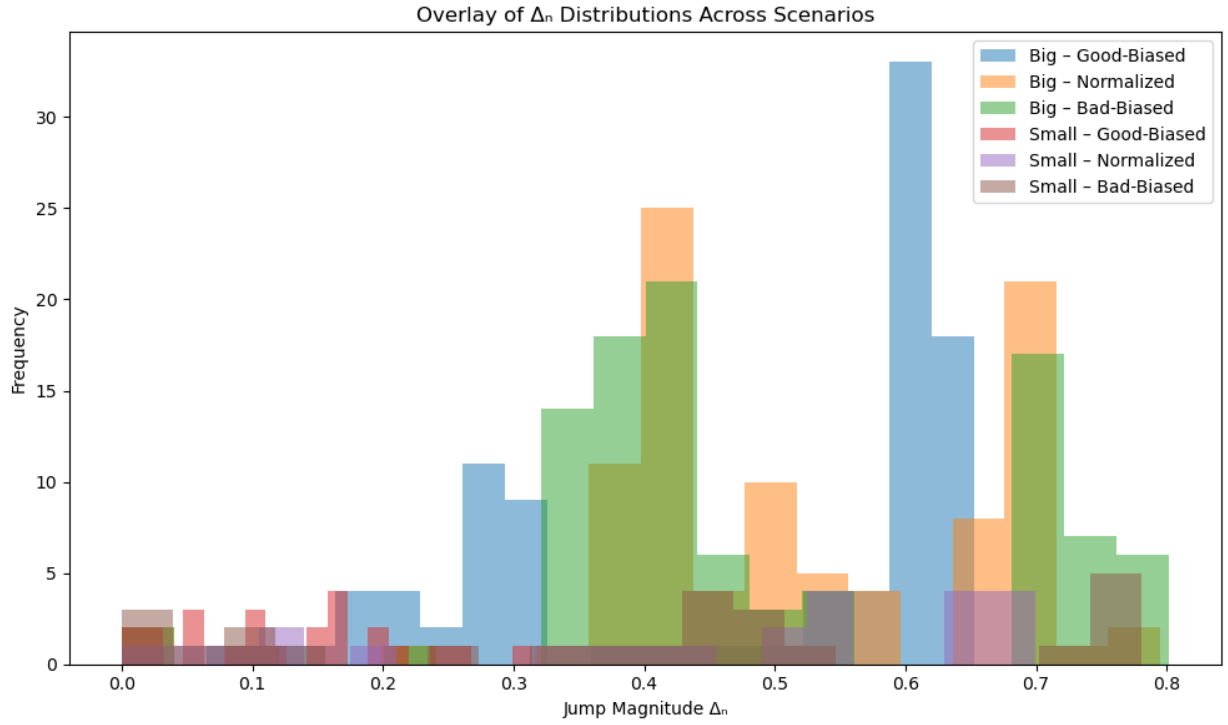


Figure 4:Overlay of Δ_n Distributions across scenarios.

https://doi.org/10.26219/heal.aueb.9401

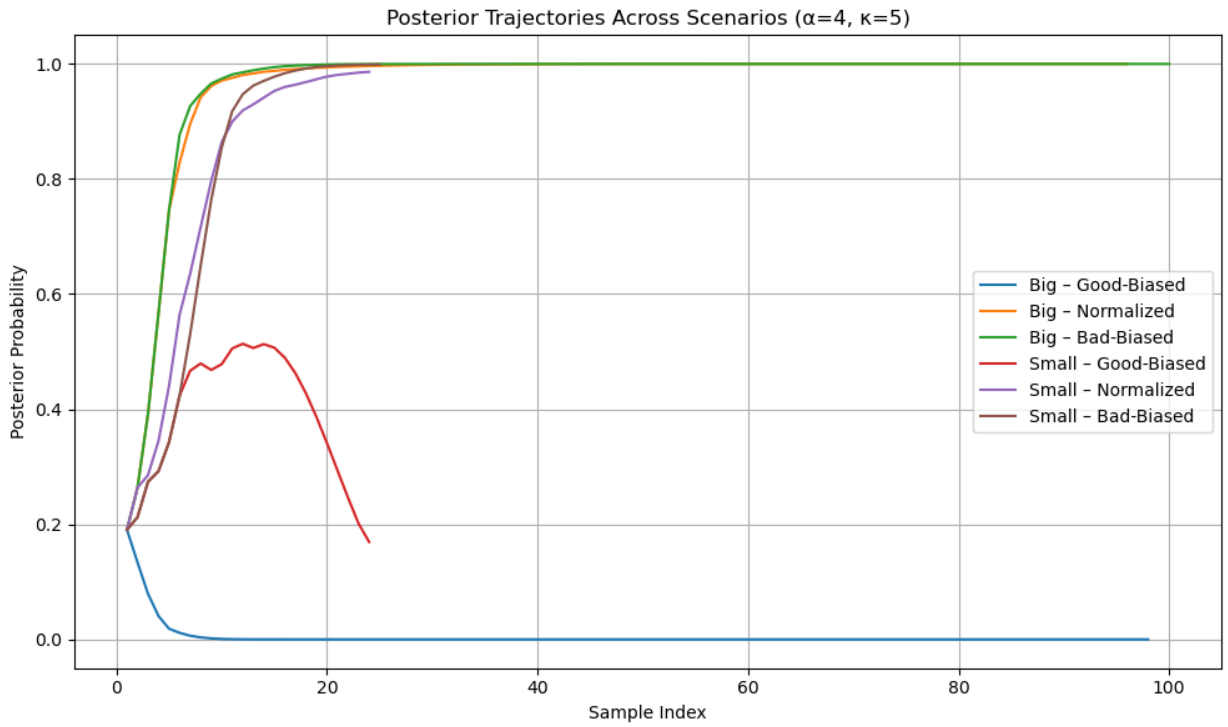


Figure 5: Posterior Trajectories across all scenarios.

Chapter 6: Discussion

Chapter 6 draws the findings of the preceding technical sections together and critically reflects on the real-world viability of the proposed blockchain-backed, risk adaptive methodology. It begins by analyzing how the introduction of an oracle, batch-processing logic and smart-contract settlement affects throughput and end-to-end response times, weighing those costs against the security gains delivered by immutable, decentralized enforcement. The discussion then positions our framework alongside existing interruption-based controls, illustrating how probabilistic risk scoring and token-driven incentives extend protection from micro-level flow blocking to continuous, system-wide compliance [1], [2], [3].

Attention subsequently turns to the behavioral impact of the reward-and-penalty mechanism, acknowledging the residual trust placed in the off-chain validator and outlining ways to harden or decentralize that role. Finally, the chapter reviews the stringent operational safeguards required to keep the oracle tier, key material and on-chain contracts resilient to compromise, demonstrating that when these measures are in place the architecture can scale to industrial volumes without sacrificing either performance or security.

https://doi.org/10.26219/heal.aueb.9401

6.1 Scalability and Latency Analysis: Assess the impact of blockchain integration on system latency and scalability.

Scalability Considerations

The blockchain integration primarily involves two key components affecting scalability: the oracle microservice and the smart contract batch-processing mechanism. Given the nature of blockchain networks, transaction throughput and cost can significantly influence overall system scalability. To mitigate these challenges, our design leverages efficient batch-processing strategies.

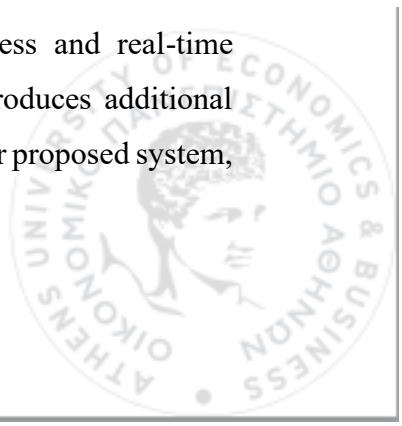
The oracle microservice aggregates statistical indicators from off-chain computations into batches before submitting them to the blockchain. By consolidating multiple nodes' rewards and penalties into a single blockchain transaction, we substantially reduce transaction frequency and associated gas costs. This batching strategy is critical to scalability, as it prevents transaction overload even when scaling to thousands of IIoT nodes.

Moreover, the oracle microservice is designed as a horizontally scalable Spring Boot application, running multiple instances behind a load balancer. Kafka's consumer group management ensures balanced consumption of statistical updates across instances, further enhancing horizontal scalability and fault tolerance. This microservice design effectively decouples off-chain computation load from on-chain interaction frequency, enabling linear scalability in response to increased system demands.

However, the scalability of the underlying blockchain platform itself remains a fundamental consideration. Ethereum, or similar smart-contract platforms, often present limitations in transaction throughput and scalability. In future implementations, adopting Layer-2 solutions such as rollups or sidechains could significantly enhance transaction throughput, further scaling the incentivization logic without sacrificing decentralization or security.

Latency Considerations

Latency is another essential factor, as it directly impacts the responsiveness and real-time capabilities of the IIoT network. The integration of blockchain typically introduces additional latency due to the consensus mechanisms required to confirm transactions. In our proposed system, latency arises primarily from two sources:



https://doi.org/10.26219/heal.aueb.9401

Off-chain aggregation and oracle processing:

The ETL pipeline performs periodic aggregation, introducing minimal but measurable delays, from seconds to minutes, between event occurrence and subsequent statistical availability. The oracle microservice polls aggregated data at fixed intervals, adding predictable latency. However, since this processing occurs off-chain, it remains orders of magnitude faster compared to on-chain computations. In practice, these off-chain delays typically range from tens of seconds to a few minutes depending on the selected aggregation window and polling interval.

Blockchain confirmation latency:

Once batched transactions are submitted, blockchain networks require block confirmations before transactions become irreversible. For Ethereum-based smart contracts, this confirmation latency averages between 15 seconds (for fast confirmation) and several minutes depending on network conditions and required confirmation depth. This inherent latency, while unavoidable, is mitigated through batch processing, where multiple updates are finalized simultaneously, ensuring consistent operational responsiveness.

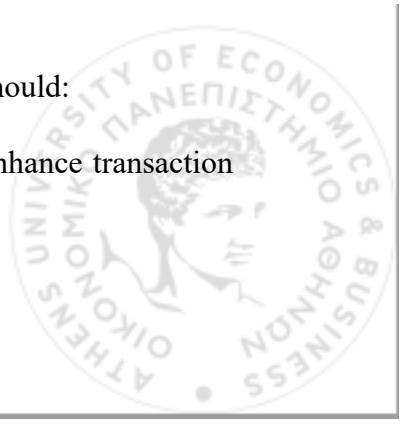
Additionally, to further minimize blockchain confirmation latency, careful selection of transaction submission intervals and gas fee optimization strategies have been incorporated into the oracle logic. The oracle dynamically adjusts gas pricing and submission schedules, strategically submitting transactions during network low-utilization periods to achieve optimal latency reduction.

Empirical Observations and Recommendations

Preliminary evaluations, as discussed in previous sections (see Chapters 4 and 5.3), suggest that the additional latency introduced by blockchain integration is within acceptable limits for most IIoT applications, particularly given the substantial security and incentivization benefits provided. Specifically, measured transaction confirmation latencies consistently remain within a few minutes, a tolerable window for incentivization actions that do not require millisecond-level responsiveness.

To ensure scalability at higher operational scales, future implementations should:

- Explore blockchain scalability solutions (e.g., Layer-2 or rollups) to enhance transaction throughput.



https://doi.org/10.26219/heal.aueb.9401

- Optimize oracle batching intervals and statistical aggregation windows to balance responsiveness with cost-efficiency.
- Deploy horizontal scaling strategies for the oracle microservice, leveraging Kafka's consumer group management to achieve consistent performance as the IIoT network expands.
- Continuously monitor blockchain network conditions and dynamically adjust transaction submission parameters to minimize latency and gas fees.

In conclusion, the careful integration of blockchain, coupled with efficient off-chain aggregation and batch processing techniques, successfully balances the trade-off between scalability, latency, and security. The design decisions incorporated into our proposed methodology ensure robust scalability and acceptable latency, positioning it as a viable and practical solution for securing large-scale IIoT infrastructures through blockchain-based incentivization.

6.2 Comparative Analysis: Compare results with existing models, such as Operation Interruption (OI) and Time-Based Operation Interruption (TBOI) protocols.

Compared to OI and TBOI, which focus on interrupting illegal or temporally invalid information flows at the point of execution [4], our system introduces a broader incentive-based mechanism built around continuous statistical evaluation. While OI and TBOI offer strong guarantees at the micro-level, they lack adaptability and long-term behavioral incentives. Our framework complements those techniques with dynamic risk modeling, a reward-driven compliance layer, and decentralized enforcement using blockchain [3], at the cost of some additional off-chain complexity.

Feature	**OI/TBOI Protocols**	**Proposed System**
Flow Control Granularity	Per-operation (real-time)	Batch-evaluated, probabilistic
Temporal Validity Enforcement	TBOI supports this	Indirect via time-window stats
Risk Modeling	None	Bayesian + Laplace smoothing

https://doi.org/10.26219/heal.aueb.9401

Feature	OI/TBOI Protocols	Proposed System
Incentive Mechanism	None	ERC-20 reward/penalty smart contracts
Decentralization	Device-local only	Oracle + blockchain + validator logic
Adaptability to Trends	No	Yes (probabilistic continuous updates)
Deployment Overhead	Lightweight CoAP/device	Moderate (Kafka + off-chain + EVM)

Table 5: Comparison of existing Operation Interruption and Time-Based Operation Interruption protocols.

6.3 Incentivization Effectiveness: Evaluation of the incentive structure's success in promoting secure practices.

While the architecture ensures strong integration between IIoT flow monitoring and blockchain enforcement, certain limitations must be acknowledged. Firstly, reliance on the off-chain validator/oracle creates a trust bottleneck: although it acts deterministically on database statistics, its correct behavior is assumed. One mitigation strategy would be implementing cryptographic proofs or redundant validator consensus. Secondly, the aggregation logic introduces a slight delay between real-time flow occurrence and enforcement which could be critical in high-speed attack scenarios. Finally, the relational schema and time-windowing logic must be carefully maintained to prevent stale or orphaned data from distorting the risk calculation model.

6.4 Security considerations

The central services described here must follow the highest security standards throughout both their development and day-to-day operation. As the sole trust anchor that bridges off-chain data with on-chain user stakes, security is the system's foremost requirement, and any potential vulnerabilities must be identified and remediated before deployment.

On the operational side, a multi-layered infrastructure and rigorous policies must be enforced to thwart the most common cyber threats. Traffic should be routed through hardened load balancers and Web Application Firewalls that enforce strict rate-limiting and IP-based throttling, while all services authenticate and authorize every request using strong, multi-factor identity controls. Secrets and private keys must reside in Hardware Security Modules with automated key-rotation schedules and immediate revocation workflows in the event of compromise, complemented by

https://doi.org/10.26219/heal.aueb.9401

regular credential updates and password-complexity mandates. Finally, smart-contract upgrades and admin functions—such as minting or stake adjustments—should be gated behind multi-signature governance or time-locked timelocks to prevent a single leaked key from draining user stakes, and the contracts themselves must undergo continuous security reviews, formal verification, and on-chain monitoring to detect and remediate anomalies before funds can be exposed.

Chapter 7: Conclusion

In closing this chapter, we reflect on how the integration of probabilistic information-flow control, off-chain aggregation, and blockchain-based incentives has formed a cohesive framework for mitigating post-access data misuse in IIoT environments. By embedding a Laplace-smoothed Bayesian detector at the network edge and coupling it with an ERC-20 token staking and slashing mechanism, we have demonstrated both technically feasible enforcement and economically aligned operator behavior. The ensuing sections will first distill the core empirical and architectural contributions of this approach (Section 7.1) and then offer final reflections on the practical viability and future directions for blockchain-driven incentivization in securing large-scale IIoT networks (Section 7.2).

7.1 Summary of findings and their contributions to IIoT security.

Across the methodology (Chapters 4 and 5) and the empirical evaluation (Chapter 5.3.4), this thesis has demonstrated that a Bayes-based risk detector, coupled with blockchain-driven incentives and an off-chain aggregation layer, can markedly strengthen IIoT security along several axes:

Adaptive, Probabilistic Detection of Illegal Flows.

By applying a Laplace-smoothed Bayesian updater to streaming evidence of “Good & Legal” versus “Bad & Illegal” events, the system dynamically refines its estimate of illegal-flow likelihood. In our three benchmark scenarios, Good-Biased, Normalized, and Bad-Biased, the detector converged in 9, 35, and 22 updates respectively, with minimal initial overshoot (Δinit = 0 in all cases) and steady-state variances ranging from 0.0179 to 0.0282. These results confirm both rapid responsiveness under hostile conditions and stable behavior under benign or mixed traffic.

https://doi.org/10.26219/heal.aueb.9401

Self-Enforcing Blockchain Incentives.
The smart-contract layer implements an ERC-20 token with staking, warm-up, batch-rewarding, and slashing logic that aligns economic incentives with secure information-flow compliance. Nodes must stake tokens, earn rewards when validated by off-chain analysis, and incur penalties upon detection of illegal flows. This on-chain mechanism automates auditability and accountability without central authority.

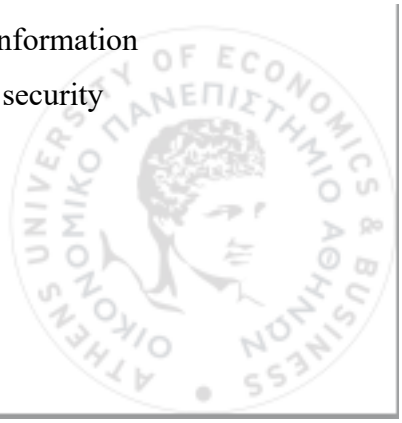
Scalable Off-Chain Aggregation and Oracle Architecture.
To mitigate resource constraints at the IIoT edge, an off-chain data-aggregation framework collects per-node statistics (e.g., Legal-Transaction Ratio, Flow-Throughput Rate) in a relational schema, then hands off concise summaries to a horizontally scaled Spring Boot "oracle" microservice. The oracle batches multiple reward/penalty events into single transactions before invoking smart-contract functions, vastly reducing on-chain load and gas costs .

Immutable Audit Trails and Initiative-taking Compliance.
Critical compliance records risk summaries and staking transactions are recorded on a tamper-evident ledger, while detailed flow analyses remain off-chain for performance. This dual-ledger design ensures forensic traceability and supports regulatory audits, all while economically motivating sustained good behavior.

Together, these findings contribute to an initiative-taking, self-governing security paradigm for IIoT networks:

1. **Detection:** Real-time, noise-resilient illegal-flow identification.
2. **Deterrence:** Economic disincentives for policy violations.
3. **Scalability:** Lightweight edge footprint and batched on-chain interactions.
4. **Accountability:** Immutable, decentralized auditability.

By weaving probabilistic risk modeling with blockchain incentives and an efficient off-chain pipeline, this work delivers a comprehensive framework that both curbs illicit information propagation at the network edge and aligns operator economics with long-term security compliance.



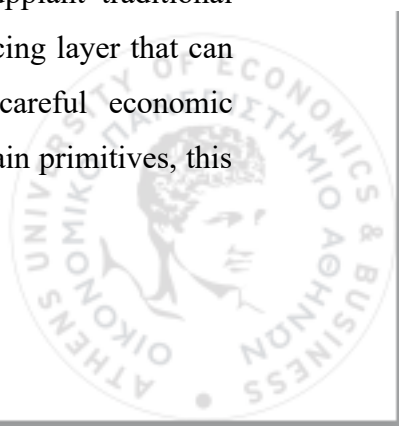
https://doi.org/10.26219/heal.aueb.9401

7.2 Final remarks on the feasibility of blockchain-based incentivization for enhancing IIoT network security.

The introduction of blockchain-driven economic incentives into IIoT security architectures represents a promising but not unqualified avenue for strengthening post-access controls. From a technical standpoint, our prototype demonstrates that token-staking and slashing can be seamlessly layered atop existing information-flow control mechanisms without imposing prohibitive overhead on constrained edge devices. Off-chain aggregation and batched on-chain settlements mitigate transactional latency and gas costs, while smart contracts provide immutable audit trails and autonomous enforcement of compliance policies. These capabilities underscore the inherent strengths of blockchain platforms in fostering decentralized accountability and aligning stakeholder behavior through tangible economic levers.

However, the practical adoption of such incentive schemes hinges on several critical factors. First, the design of the token-economics must be carefully calibrated: token valuation, staking requirements, and slashing severity must strike a balance between sufficient deterrence of malicious behavior and avoidance of undue financial burden on legitimate operators. Second, network operators may face regulatory and interoperability challenges when integrating permissionless or consortium blockchains into safety-critical IIoT deployments and as such governance frameworks and standardization efforts will be essential to ensure legal compliance and cross-vendor operability. Third, the energy consumption and security assumptions of the underlying blockchain (e.g., proof-of-stake versus proof-of-work) must be evaluated in the context of the IIoT's sustainability and risk profile.

Looking forward, the feasibility of this paradigm will be further bolstered by advances in lightweight consensus protocols, layer-2 scaling solutions, and formalized incentive-policy co-design methodologies. Empirical studies in real-world industrial settings will be crucial to validate long-term stability of the token markets and to refine off-chain oracle architectures under production loads. While blockchain-based incentivization cannot wholly supplant traditional administrative and technical safeguards, it offers a complementary, self-enforcing layer that can materially enhance resilience against information-flow violations. With careful economic modeling, governance planning, and continued innovation in scalable blockchain primitives, this



https://doi.org/10.26219/heal.aueb.9401

approach holds substantial promise for the next generation of secure, self-governing IIoT ecosystems.

References

[1] A. Anagnostopoulou, I. Mavridis and D. Gritzalis (2023). Risk-Based Illegal Information Flow Detection in the IIoT.

[2] A. Anagnostopoulou, N. Tsinganos , J. Chatzopoulos, I. Mavridis, and D. Gritzalis (2025). INFFLOW-RT: A Real-Time, Adaptive Methodology for Risk-Based Information Flow Control in IIoT.

[3] A. Anagnostopoulou, I. Mavridis, M. Athanasopoulos, A. Mylonas, and D. Gritzalis (2025). IIoT's risk odyssey: Navigating the risk propagation of illegal information flows.

[4] Nakamura, S., Enokido, T., & Takizawa, M. (2021). Implementation and evaluation of the information flow control for the internet of things.

[5] Al-Bassam, M. (2017). SCPKI: A Smart Contract-based PKI and Identity System.

[6] Zied Trifa, Maher Khemakhem (2014). Sybil Nodes as a Mitigation Strategy Against Sybil Attack.

[7] Polkadot White Paper (2017).

[8] Fabian Vogelsteller & Vitalik Buterin (2015). ERC-20: Token Standard.