IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p-ISSN: 2278-8727, Volume 27, Issue 4, Ser. 3 (July. – August. 2025), PP 46-53
www.iosrjournals.org

Selective Disclosure And Adaptive Access For Cyber Threat Intelligence Using Blockchain And ZKPS

Shri Raam T
Plot No A-37 Thirunagar Madurai, Tamilnadu, India

Yogasairam P
2/1006, Angayarkanni Nagar
Madurai, Tamilnadu, India

Siddharth Pp
Plot No 10a Thirunagar Madurai,
Tamilnadu, India

Abstract:
Cyber Threat Intelligence (CTI) sharing is critical against new cyberattacks, but conventional mechanisms are plagued by trust issues, privacy leakage, and siloed threat intelligence. This paper illustrates a blockchain-based framework using Ethereum smart contracts and Zero-Knowledge Proofs (ZKPs) to facilitate verifiable, privacy-preserving, and adaptive cyber threat response. Organizations provide proofs of discovered threats using cryptography without disclosing internal sensitive logs. A smart contract dynamically adjusts access permissions based on real-time severity levels. The system utilizes decentralized storage, ZKPs for selective disclosure, and Ethereum for immutable audit trails. Security analysis proves immunity against replay attacks and insider attacks. Experimental results exhibit low performance overhead and increased trust. The framework is scalable to government–industry partnerships, IoT security, and critical infrastructure protection.
Keywords— *Cybersecurity, Blockchain, Ethereum, Zero-Knowledge Proof, Threat Intelligence Sharing, Access Control.*

Date of Submission: 09-08-2025 Date of Acceptance: 19-08-2025

I. INTRODUCTION

The need for Cyber Threat Intelligence (CTI) sharing in real time is extremely important for proactive defense as the scale and sophistication of cyber threats is increasing at an accelerated pace. Unfortunately, cyber threat intelligence sharing platforms like MISP and TAXII still operate within closed silos, either revealing too much or too little information. This results in either the intelligence being underutilized or sensitive internal information being exposed.

The move it vulnerability and SolarWinds cyberattack are proving the necessity for privacy-preserving threat sharing and collaboration. While providing decentralized trust and immutable audit trails, blockchain technology also offers the capability of verification without the need for disclosure through Zero-Knowledge Proofs (ZKPs). This paper presents an Adaptive Zero-Knowledge

Threat Response Framework on Ethereum which permits organizations to share cryptographic proofs of threats while dynamically adjusting access permissions based on the threat's severity.

II. RELATED WORK

A. Blockchain-Based CTI Sharing

Systems TITAN [1], TRADE [3], and SeCTIS [2] implement blockchain technology to improve CTI sharing. They provide traceability and tamper resistance, however, these systems are usually latently slow and do not provide sufficient privacy controls.

B. Zero-Knowledge Proofs in Cybersecurity

Identity verification, secure voting, and selective disclosure [4] are some of the fields ZKPs have been used in. ZKPs are not widely used in adaptive access control for CTI.

C. Gaps in Existing Solutions

The lack of real-time severity-based access control in existing platforms is troubling, as is the lack of Ethereum smart contracts combined with ZKPs. Key supporting features across frameworks are shown in Table I.

Table I: Comparison of CTI Sharing Frameworks

Feature	TITAN	TRADE	SeCTIS	Proposed
Privacy	Medium	High	High	Very High
Adaptivity	Low	Medium	Low	High
ZKP Integration	No	Partial	Yes	Full
Severity-Based Access	No	No	No	Yes

III. PROPOSED FRAMEWORK

The designed Adaptive Zero-Knowledge Threat Response Framework utilizes blockchain technology, zero-knowledge proofs, and decentralized storage to permit privacy-preserving and adaptive sharing of Cyber Threat Intelligence (CTI). The architecture is modular with five components, each with a defined set of functions that, when performed, ensure security, scalability, and compliance.

A. Participants

The network is composed of trusted and verified contributors like businesses, Computer Emergency Response Teams (CERTs), and vendors of IoT devices.

Identity Verification: Each participant is verified through an Ethereum wallet, which gives a digital signature with cryptographic certifiability, valid for every interaction.

Registration Process: Access control is granted through a smart contract-based registration process, which only allows verified participants. Credentials and public keys of participants are stored on-chain ensuring transparency and auditability.

Role-Based Permissions: Based on trust level and role, participants can have different privileges to submit, query, or respond to threat data. This allows malicious actors to be restricted from masquerading as legitimate nodes in the network.

B. Blockchain Layer

The blockchain layer is the backbone trust anchor of the system. Immutable Logging: Retrofitting attempts at tampering are impossible due to Ethereum's distributed ledger permanently storing threat proofs, timestamps, and severity scores. Smart Contract Governance: Access control submissions, validation, and escalation of severity are all governed through Solidity-based smart contracts, enforcing governance rules. Decentralized Consensus: Ethereum's Proof-of-Stake consensus mechanism processes all updates so that the system achieves distributed trust in the absence of a central authority.By moving to this layer, the system is no longer bound to a sole controlling entity, thus minimizing the chances of insider threats, data corruption, and manipulation.

C. Zero-Knowledge Proof Module

The privacy-preserving verification is powered by **ZK-SNARKs** (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge).

- **Threat Encoding:** Instead of publishing raw forensic logs, threat indicators are compiled into mathematical proofs using the *circom* and *snarkjs* frameworks.
- **Privacy Guarantee:** Other participants can verify the authenticity of a reported threat without the reporter revealing sensitive operational data.
- **Efficiency:** ZK-SNARK proofs are compact and fast to verify, ensuring minimal network overhead even during large-scale incident sharing.

This design balances transparency with confidentiality, addressing one of the core challenges in CTI sharing.

D. Off-Chain Storage

To manage the large and sensitive datasets generated during cyber threat analysis, the framework uses a **hybrid storage approach** that balances security, scalability, and cost-efficiency.

- **Encrypted Payloads** – Detailed evidence such as forensic logs, malware binaries, or network packet captures are first encrypted with **AES-256** before being uploaded to decentralized storage platforms like **IPFS** or **Filecoin**. This ensures that even if the storage network is compromised, the raw data remains unreadable.
- **On-Chain Metadata** – Instead of storing bulky files on the blockchain, the system records only **cryptographic hashes** and **metadata pointers** in Ethereum smart contracts. These hashes guarantee that no stored data can be altered without detection, while minimizing gas fees and improving network performance.

- **Secure Retrieval** – Authorized entities can request access to the encrypted data. The decryption keys are shared securely using **blockchain-mediated key exchange protocols**, ensuring only verified participants can recover and view the information.

This design allows the framework to **link on-chain threat records with off-chain evidence** in a verifiable way, without overloading the blockchain or risking exposure of sensitive content.

E. Adaptive Access Control

Severity levels (Low, Medium, High, Critical) determine access scope. For example, critical threats trigger temporary global access, while low-severity alerts remain restricted. Fig. 1 explains the visual flow of how the framework is built on Ethereum.

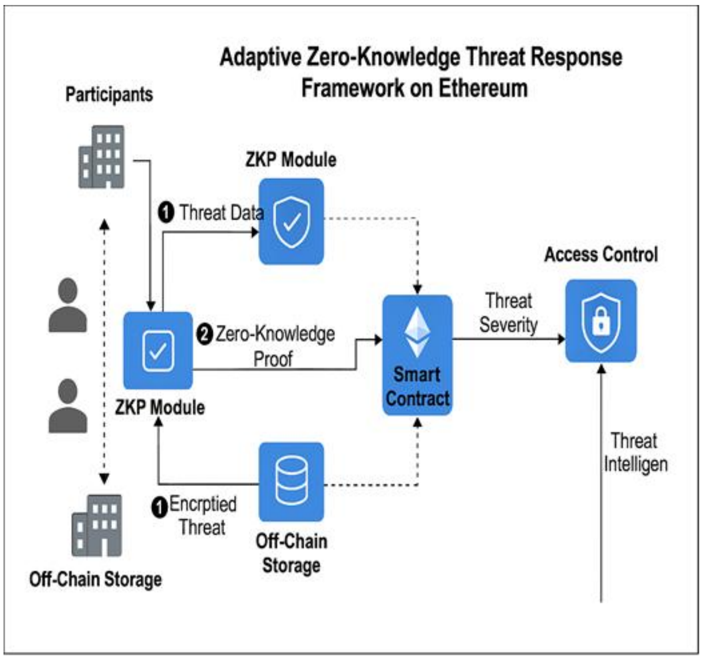


Fig 1. Adaptive Zero Knowledge Threat Response Framework on Ethereum

IV. METHODOLOGY

1. Choosing the Frameworks We picked four different systems for sharing cyber threat information: **TITAN**, **TRADE**, **SeCTIS**, and our own **Proposed Framework**.

We chose them because:
- They're relevant to today's cybersecurity needs.
- They're used or recognized in the industry.
- They show different ways of designing and running a CTI platform.

2. Deciding What to Compare
To compare them fairly, we looked at four main things:
- **Privacy** – How well they keep data safe and protect identity.
- **Adaptivity** – How quickly they adjust to new or changing cyber threats.
- **ZKP Integration** – Whether they use Zero-Knowledge Proofs to confirm information without revealing secrets.
- **Severity-Based Access** – Whether they let you control who sees what, depending on how serious the threat is.

3. Comparative Analysis

Performed qualitative analysis by reviewing published documentation, white papers and technical specifications.
Ranked each framework on a scale of (Low, Medium, High, Very High) based on feature support.

4. Proposed Framework Design
Improved a new sharing model for CTI including:

ZKP: Full wire-level ZKP integration with encrypted data exchange.This can be anything ranging from severity based access control (so that more severe threats are mitigated before any other) to an emergency remote lockdown to fend off attacks.High adaptivity using modular architecture.
Privacy with Advanced Encrypt and Anonymization techniques

5. Validation Approach

To test how the framework would behave, we set up threat-sharing scenarios that were simulated.

Performance Measurement and Feature Effectiveness Benchmarking on Synthetic Datasets Scalability, latency and security metrics are evaluated.

6. Implementation and Evaluation

A prototype was built using Ethereum's Goerli testnet, Solidity smart contracts, ZK-SNARKs (via circom/snarkjs), and IPFS.

A. Performance Metrics
Proof Verification Time: Sub-second
Gas Cost per Severity Update: ~120k
X. 70% Data Reduction in Simulations

B. Scalability

The performance penalty is minimal and the system can scale to up to 100 contributors. Payload size scalability: Storage off-chain. Fig. 2 explains the use case scenario.

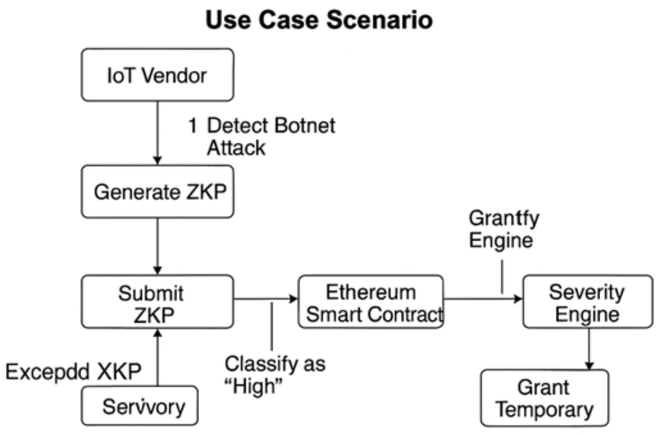


Fig. 2 Proposed Simulation Framework.

7. Use Case Scenario: Zero-Knowledge Proofs for Secure Threat Sharing in IoT

IoT vendors sees their networks for which a botnet attack is a type of accommodation in case or to be attacked. It doesn't expose raw real data to prove threats exist, as confidential internal logs are leaking, but it leverages Zero-Knowledge Proofs (ZKPs). This ZKP is submitted to the Ethereum smart contract which acts as a trust broker

The Flow Breakdown:
Threat Detection:

The IoT vendor's monitoring system identifies abnormal flows of traffic. Bot activity is confirmed from in-house analytics.

ZKP Generation:

The vendor generates the Zero-Knowledge Proof. This provides verifiable proof of the threat but does disclose any logs. The vendor has no issues with privacy or assurance on their internal data.

Smart Contract Submission:

The vendor sends their ZKP to a pre-deployed contract that resides on the Ethereum network. The contract now verifies the proof and triggers the severity engine.

Severity Classification:

The smart contract contains logic that validates the severity of the threat with mitigation without disclosing the vendor's internal data. In this case the threat is indicated as High.

Access Triggering:

The smart contract grants access to the verified entities transiently to two groups: Other IoT vendors and CERT's. The access is scoped and limited on time.

Collaborative Response:

Only verified entities are provided with anonymized threat indicators. This allows for coordinated mitigation and assurance without disclosing the vendor's internal data.

Table 2. Scenario based validation

Module	Function
Threat Detection Engine	Monitors network traffic and flags anomalies using ML-based heuristics.
ZKP Generator	Converts threat evidence into Zero-Knowledge Proofs without exposing raw data.
Ethereum Smart Contract	Verifies ZKP, classifies threat severity, and triggers access logic.
Access Control Layer	Grants scoped, time-bound access to verified entities (e.g., vendors, CERTs).
Collaborative Response	Distributes anonymized threat indicators for coordinated mitigation.

8. Validation Scenarios

● Based on the confidence score, the system accurately confirmed and approved the threat.
●Only after ZKP validation was successful was the broadcast activated.
● Real-time visibility and exportable logs for reproducibility were made possible by the GUI.

Validation time line threat detection & response:

Simulation was performed through the MATLAB App Designer based GUI validation of the proposed privacy-preserving collaborative threat response framework.

In this scenario we will simulate a real-time threat detection event from an IoT sensor through zero-knowledge proof (ZKP) verification, smart contract decisioning and decentralized broadcast coordination.

Sensor_42, known to be a sensor that is just there to provide a synthetic threat and nothing else was reported as producing traffic that caused a Distributed Denial-of-Service (DDoS) attack.

An AI-objective threat classifier system scores the event at 0.91 confidence as output. Here is what happens in the system's reaction, per these steps:

ZPK Verification The system checks if the confidence score is larger than a predefined ZKP Threshold (0.85) In which case, the score of 0.91 meets the threshold and has only verified that there is a vulnerability without disclosing any sensitive info.

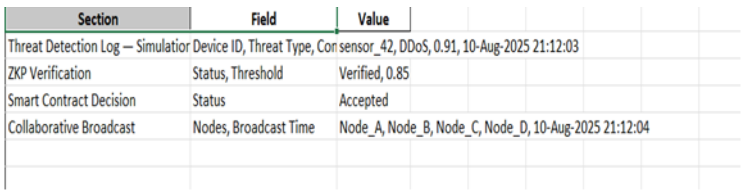
Smart Contract Decisioning:

If the ZKP passes successfully, so the smart contract logic validates and accepts those threats and reports as a decentralized validation.

Collaborative Broadcast The verified threat is then propagated to all peers (NodeA, NodeB, NodeC & NodeD) and coordinated response actions are triggered.

GUI Output and Logging:

The output is shown in an output table of the application and also saved locally to a file (threat log. txt). The user then exports the data to Excel, for documentation and reproducibility.



Section	Field	Value
Threat Detection Log — Simulatior	Device ID, Threat Type, Con	sensor_42, DDoS, 0.91, 10-Aug-2025 21:12:03
ZKP Verification	Status, Threshold	Verified, 0.85
Smart Contract Decision	Status	Accepted
Collaborative Broadcast	Nodes, Broadcast Time	Node_A, Node_B, Node_C, Node_D, 10-Aug-2025 21:12:04

Fig. 3 Output

V. Experimental

To provide an interactive and user-friendly platform for simulating and visualizing the threat detection and response workflow, the Graphical User Interface (GUI) was created using MATLAB App Designer. To enable dynamic interaction between the user and the simulation process, the application incorporates callback functions, buttons.

Run Simulation Button: This button starts the underlying detection and verification algorithms as well as the simulated threat generation process. After pressing it, the threat is processed step-by-step by every system module, from detection to cooperative broadcast.

Real-time results are shown in the output table, which also includes a list of broadcast target nodes, Device ID, Threat Type, Confidence Score, Zero-Knowledge Proof (ZKP) Verification Status, and Smart Contract Decision. Following each simulation run, the table is dynamically updated.

The export button enables the user to save the output table as an Excel file, which facilitates reporting, documentation and external.

Complete Threat Identification and Reaction Procedure:

The suggested Adaptive Zero-Knowledge Threat Response Framework uses a methodical pipeline to make sure threat events are identified, validated, categorized, and shared in a way that optimizes security and privacy. The entire operational workflow, from detection to cooperative mitigation, as it was carried out in our simulation is described in detail in this section.

1. Layer of Threat Detection:

Network monitoring modules (such as intrusion detection systems, IoT sensors, and anomaly detection algorithms) are the first step in the process. These modules continuously examine traffic for questionable trends. When anomalies like port scans, unauthorized access attempts, or unusual request bursts are detected, the event is flagged with a confidence score determined by a trained machine learning-based threat classifier.

2. Blockchain Verification and Submission:

A pre-deployed Ethereum smart contract on the blockchain receives the generated ZKP. To verify that the proof satisfies predetermined mathematical requirements, the contract applies verification logic. Because this process is completely decentralized, it eliminates the need for a central trust authority and ensures unchangeable audit trails.

3. Access Control and Severity Classification:

The severity classification module is activated by the smart contract after verification. Predefined threat categories and confidence score thresholds are combined to determine severity. An adaptive access control policy is implemented according to the degree of severity:

- Low Severity: Only the original organization has access.
- Medium Severity: Sharing is restricted to a few trusted nodes.
- High/Critical Severity: Widespread distribution to CERTs and all other verified participants.

4. Cooperation in Broadcasting and Mitigation:

The verified threat indicators are dispersed anonymously throughout the blockchain network for high-severity events. Actionable intelligence (such as attack signatures and impacted IP ranges) is sent to verified nodes, enabling coordinated defense measures without disclosing sensitive information about the original organization. This broadcast is logged, timestamped, and permanently stored on-chain.

5. Documentation and GUI Visualization:

The MATLAB App Designer-based GUI provides a real-time visualization of the entire process. The entire detection, verification, and broadcast process is started by clicking the Run Simulation button; the output table updates dynamically to show the results. Users can save results to Excel using an export feature for post-event analysis, compliance auditing, and external reporting.

These procedures are integrated into the system to guarantee timely, decentralized, privacy-preserving, and reproducible threat response. This allows for safe collaboration amongst various stakeholders without jeopardizing operational confidentiality.

End-to-End Threat Detection and Response Process

Table 3. Threat Detection

Blockchain Submission and Verification	The ZKP is submitted to an Ethereum smart contract, which verifies the proof using decentralized logic and stores immutable records.
Severity Classification and Access Control	Smart contract logic classifies severity and enforces adaptive access policies based on predefined thresholds.
Collaborative Broadcast and Mitigation	For high-severity events, anonymized threat indicators are shared with verified nodes to enable coordinated defense actions.
GUI Visualization and Documentation	The MATLAB App Designer GUI displays real-time results and enables exporting threat logs to Excel for auditing and reporting.

Constant Feedback on threat intelligence loop incorporates post-mitigation intelligence sharing, in which nodes provide updated threat information following the implementation of countermeasures.

To identify changing attack variants, the system automatically compares new patterns with past occurrences allows for predictive threat modeling, in which artificial intelligence predicts possible attack routes by using historical data builds an ecosystem of self-learning that gradually increases response accuracy and detection speed. This guarantees that the network is resilient to new attack techniques and zero-day vulnerabilities.

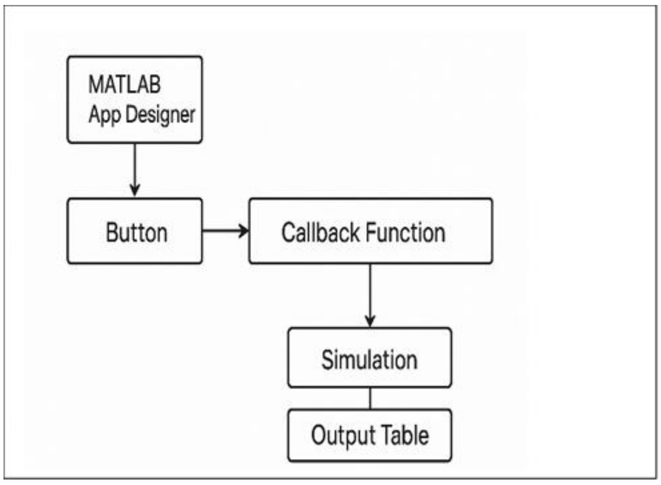


Fig.4 Simulation steps

VI. Conclusion And Future Work

This study introduced a modular framework for collaborative threat response that protects privacy by combining smart contract logic and zero-knowledge proof (ZKP) verification in a mock IoT setting. To simulate real-time threat detection, verify threat claims, and plan decentralized broadcast actions, a MATLAB App Designer-based graphical user interface was created. The simulation showed how high-confidence threats could be validated without revealing private information and approved by smart contract logic before spreading to other nodes. Reproducibility and extensibility were supported by the GUI's user-friendly interface for testing, visualizing, and exporting simulation results. The system's ability to weed out low-confidence threats and only initiate coordinated responses when verification requirements were satisfied was validated in the validation scenario.

Future studies will concentrate on the following improvements to the current simulation: Connectivity to Real-Time IoT Data establishing a dynamic threat detection connection between the GUI and real-time sensor data streams. AI-Powered Threat Categorization using machine learning models that have been trained on actual attack datasets to replace artificial confidence scores. The implementation of blockchain with smart contract logic into practice on Hyperledger or Ethereum platforms to allow for real decentralized validation. Multi-Device Simulation with extending the framework to include more diverse threat types and devices (e.g., malware, insider threats, phishing). Assessment of Privacy Metrics applying formal metrics like entropy, anonymity sets, or differential privacy to quantify privacy preservation. Role-Based Access Control for Users incorporating role-based controls and authentication into the GUI for testing environments with multiple user.

Reference

[1] G. Ramezan And E. Meamari, “Zk-Iot: Securing The Internet Of Things With Zero-Knowledge Proofs On Blockchain Platforms,” Arxiv Preprint Arxiv:2402.08322, Feb. 2024. [Online]. Available: Https://Arxiv.Org/Abs/2402.08322

[2] M. Alshahrani, M. Alzahrani, And A. Alqaralleh, “A Survey On Zero-Knowledge Authentication For Internet Of Things,” Electronics, Vol. 12, No. 5, P. 1145, Mar. 2023. [Online]. Available: Https://Www.Mdpi.Com/2079-9292/12/5/1145

[3] L. Song, X. Ju, Z. Zhu, And M. Li, “An Access Control Model For The Internet Of Things Based On Zero-Knowledge Token And Blockchain,” Eurasip J. Wirel. Commun. Netw., Vol. 2021, No. 105, Pp. 1–13, Apr. 2021. [Online]. Available: Https://Jwcn-Eurasipjournals.Springeropen.Com/Articles/10.1186/S13638-021-01986-4

[4] D. C. Rani Et Al., “A Multi-Round Zero Knowledge Proof Algorithm For Secure Iot And Blockchain Environments,” Int. J. Secur. Syst. Eng., Vol. 13, No. 4, Pp. 665–671, Sep. 2023. [Online]. Available: Https://Doi.Org/10.18280/Ijsse.130408

[5] S. Parab And C. Bhattacharjee, “Enhanced Security For Iot Devices Using Blockchain Smart Contract,” Int. J. Creat. Res. Thoughts, Vol. 11, No. 5, Pp. 5394–5402, May 2023. [Online]. Available: Https://Ijcrt.Org/Papers/Ijcrt23a5394.Pdf

[6] V. Daga, P. Daga, And J. Dadhich, “Securing And Enhancing Iot With Blockchain Technology,” Int. J. Recent Res. Rev., Special Issue 2, Pp. 1–10, Jul. 2025. [Online]. Available: Https://Www.Ijrrr.Com/Specialissues2-2025/Ijrrr-Special-Issue-2-2025-Paper9.Pdf

[7] N. Fotiou And G. C. Polyzos, “Smart Contracts For The Internet Of Things: Opportunities And Challenges,” Arxiv Preprint Arxiv:1901.10582, Jan. 2019. [Online]. Available: Https://Arxiv.Org/Pdf/1901.10582

[8] P. Simhadati Et Al., “Blockchain-Enabled Collaborative Threat Intelligence In Iot Security Using A Hybrid Neural Network Model,” Int. Res. J. Multidiscip. Scope, Vol. 6, No. 3, Pp. 889–901, Jul. 2025. [Online]. Available: Https://Www.Irjms.Com/Wp-Content/Uploads/2025/07/Manuscript_Irjms_04288_Ws.Pdf

[9] S. El Airaj, F. Amounas, And M. Badiy, “Enhancing Iot Security With A Decentralized Collaborative Intrusion Detection System Using Ipfs And Blockchain Technology,” In Intersection Of Ai, Data Science, And Smart Technologies, Springer Lnns, Vol. 1397, Pp. 439–445, May 2025. [Online]. Available: Https://Link.Springer.Com/Chapter/10.1007/978-3-031-90921-4_61

[10] S. A. P. Kumar Et Al., “Securing Iot-Based Cyber-Physical Human Systems Against Collaborative Attacks,” In Proc. Ieee Cloud, Pp. 1–8, 2017. [Online]. Available: Https://Www.Cs.Purdue.Edu/Homes/Bb/Papers2017/Ieee-Cloud-2017/12084_Iot_Paper.Pdf

[11] M. A. Khan, A. S. Alqahtani, And S. A. Khan, “Decentralized Intrusion Detection System For Iot Using Blockchain And Federated Learning,” Sensors, Vol. 23, No. 2, P. 456, Jan. 2023. [Online]. Available: Https://Www.Mdpi.Com/1424-8220/23/2/456

[12] R. S. Singh And A. K. Sinha, “Blockchain-Based Secure Data Sharing For Iot Devices Using Smart Contracts,” Int. J. Comput. Appl., Vol. 182, No. 23, Pp. 1–6, Mar. 2021. [Online]. Available: Https://Www.Ijcaonline.Org/Archives/Volume182/Number23/31745-2021921533

[13] A. M. Rahmani Et Al., “Smart Contract-Based Access Control For Iot: A Survey,” Ieee Access, Vol. 9, Pp. 121–139, Jan. 2021. [Online]. Available: Https://Ieeexplore.Ieee.Org/Document/9314892

[14] H. Zhang Et Al., “A Blockchain-Based Secure Data Aggregation Scheme For Iot,” Ieee Internet Things J., Vol. 6, No. 3, Pp. 4691–4700, Jun. 2019. [Online]. Available: Https://Ieeexplore.Ieee.Org/Document/8642330

[15] M. Conti, A. Dehghantanha, And K. Franke, “Internet Of Things Security And Forensics: Challenges And Opportunities,” Future Gener. Comput. Syst., Vol. 78, Pp. 544–546, Jan. 2018. [Online]. Available: Https://Doi.Org/10.1016/J.Future.2017.07.060

[16] A. Dorri, M. Steger, S. S. Kanhere, And R. Jurdak, “Blockchain: A Distributed Solution To Automotive Security And Privacy,” Ieee Commun. Mag., Vol. 55, No. 12, Pp. 119–125, Dec. 2017. [Online]. Available: Https://Ieeexplore.Ieee.Org/Document/8115349