Pak. j. life soc. Sci. (2026), 24(1): 208-218 E-ISSN: 2221-7630;P-ISSN: 1727-4915

Pakistan Journal of Life and Social Sciences
www.pjlss.edu.pk

https://doi.org/10.57239/PJLSS-2026-24.1.0018

RESEARCH ARTICLE

A Method for Applying a Directed Acyclic Graph to Ensure Resource Security in Internet of Things (IoT) Information Systems

A.R. Gazizov[1], E.A. Revyakina[2*]

[1,2]Don State Technical University, Rostov-on-Don, Russia

ARTICLE INFO

Received: Jan 18, 2026

Accepted: Apr 1, 2026

Keywords

Acyclic graph architecture
Information resource security
Internet of Things
Configuration interface
Acyclic graph model
Directed acyclic graph
Software implementation of an acyclic graph
Working with configuration
Security threats

***Corresponding Author**

ABSTRACT

This article examines the problem of ensuring information resource security in Internet of Things (IoT) systems, characterized by limited device computing resources, highly dynamic networks, and a lack of unified security standards. A method for using a directed acyclic graph (DAG) for decentralized processing and confirmation of transactions between IoT devices is proposed. Unlike traditional centralized architectures and blockchain solutions, DAG allows for load balancing between nodes, eliminating single points of failure, reducing energy consumption, and ensuring high message confirmation rates. The architecture of the developed system is described, including modules for graph management, data storage, a REST API, and network visualization and simulation. The results of a software implementation of the prototype using Python, FastAPI, and SQLite are presented, demonstrating the ability to efficiently register devices, generate transactions with confirmation of previous nodes, and construct a visual graph structure. The developed method ensures resilience to individual node compromise, scalability with increasing device count, and compliance with autonomy requirements, confirming its applicability for creating secure IoT infrastructures.

INTRODUCTION

The current stage of information technology development is characterized by the active implementation of the Internet of Things (IoT)—a system of interconnected computing devices (IoT devices) that can collect and transmit data wirelessly without human intervention. The IoT is increasingly encompassing areas of human activity: from smart homes and industrial automation to urban infrastructure and medicine. The widespread use of IoT devices is leading to an increase in the volume of transmitted information and the complexity of its routing and processing, making information security issues in such systems particularly pressing.

The architecture of most IoT networks involves limited computing resources, a low level of device security, and a lack of a unified standardization system. This creates a significant number of vulnerabilities that can be exploited by attackers to gain unauthorized access, spoof data, sabotage, or create botnets based on IoT devices. As the number of devices and the degree of their interconnectivity increases, the issue of information resource security becomes critical (Smirnov, Kuznetsov, 2024).

One promising solution to this problem is the transition to decentralized architectures, specifically the use of distributed arbitration (DAG) to organize the exchange and confirmation of information between devices. Unlike traditional centralized models, DAGs allow for load balancing, increased

system resilience, and the elimination of single points of failure. Furthermore, the DAG structure reduces latency and energy costs, making it particularly attractive for implementation in IoT environments.

The relevance of this research stems from the need to develop reliable and scalable mechanisms for protecting information exchange in IoT systems that can operate under conditions of limited computing resources and high network variability. The goal of this study is to develop a method for using a directed acyclic graph to ensure resource security in IoT information systems.

CHARACTERISTICS AND CLASSIFICATION OF THE INTERNET OF THINGS (IoT)

The Internet of Things (IoT) is a distributed network of physical devices equipped with built-in computing, communication, and environmental interaction capabilities. These devices can collect, transmit, and, in some cases, process data, interacting with each other and with external systems via internet protocols.

The components of a typical IoT system include: sensors and actuators (temperature sensors, relays, cameras, etc.); controllers or gateways that perform preliminary data processing; communication tools that support various communication protocols (Wi-Fi, Zigbee, Bluetooth, LoRa, etc.) (Arunodhai et al., 2024a); cloud or local servers that process and store data; control interfaces used by end users or automation systems.

Depending on their purpose and scale of implementation, IoT systems are classified by their application area, interaction architecture, and degree of autonomy. The IoT is highly dynamic—devices can connect and disconnect at any time, making the network structure constantly changing. Another characteristic feature is the limited resources of devices: low computing power, limited memory, and low power consumption. This imposes significant limitations on possible encryption, authentication, and access control mechanisms (Arunodhai et al., 2024b).

Specific protocols frequently used in the IoT also stand out: MQTT, CoAP, and AMQP, which are focused on minimal resource consumption but were not always designed with security in mind. All of this creates a complex challenge for designing a secure architecture and requires innovative approaches, such as using DAGs as a data transfer and validation framework.

Analysis of the Current State of Security of the Internet of Things

The challenges of securing Internet of Things resources are determined by technical, architectural, and organizational factors. One significant problem is the lack of built-in security subsystems in most devices. Furthermore, a significant portion of IoT devices are designed with minimal cost and power consumption in mind. They often forego cryptographic protocols, authentication, data integrity protection, and the use of standard login and password pairs, unencrypted traffic, and open ports (Akapyev et al., 2024).

Common threats include the following:

- Unauthorized access to devices due to the lack of reliable identification and authentication mechanisms; an attacker can gain control of the device, change its parameters, or use it as an entry point into the network;
- Data interception and forgery, especially in cases where open protocols are used (e.g. HTTP, MQTT without TLS); this is especially critical in systems where personal data or control commands are transmitted;
- DDoS attacks using IoT botnets, such as the Mirai botnet, which compromised thousands of devices with open Telnet access and used them to launch massive attacks on services;
- Firmware modification and malicious updates; in the absence of a mechanism to verify the authenticity of updates, an attacker can inject malicious code into the device that will perform hidden functions;

- Scalability and centralization issues: with a large number of devices, traditional centralized security solutions become a bottleneck, as they create single points of failure and cannot cope with the load;
- Physical access to devices. IoT devices are often located in unprotected locations—on the streets, in residential buildings, and in industrial areas—making them easy to tamper with or physically hack.
- Lack of event monitoring and logging; limited memory and computing resources prevent comprehensive logging and tracking of suspicious activity.

It's also worth noting the slow response rate to vulnerabilities. Even if a vulnerability is discovered, many devices lack firmware updates, or manufacturers discontinue support shortly after release.

All these issues make the IoT environment a fertile ground for attacks, and the damage from a successful attack can extend beyond the digital realm, affecting people's physical safety. Therefore, it is necessary to seek new approaches to ensuring information security, including those based on decentralized technologies. One such approach is the use of a directed acyclic graph (DAG) structure, which eliminates the need for central nodes and increases the system's resilience to compromise.

When designing IoT networks in the Russian Federation, it is necessary to take into account the provisions of Federal Law No. 149-FZ "On Information, Information Technologies, and Information Protection," (State Duma of the Federal Assembly of the Russian Federation, 2006) which defines the fundamentals of information protection and ensures secure interaction in information systems (Baumgartner et al., 2024). When using devices in critical infrastructure, it is necessary to comply with the provisions of Federal Law No. 187-FZ "On the Security of Critical Information Infrastructure of the Russian Federation," (State Duma of the Federal Assembly of the Russian Federation, 2017) which provides for the categorization of objects, the implementation of incident detection tools, and response measures.

Analysis of the Regulatory Framework and International Experience in Protecting Internet of Things Resources

In international practice, ISO/IEC 27001 and ISO/IEC 27002 (International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), 2022a, b) standards play a special role, providing universal approaches to information security management (Shamsutdinov et al., 2024; Yang et al., 2024). In the Russian Federation, the provisions of GOST R 56939-2024 (Federal Agency for Technical Regulation and Metrology, 2024) should be taken into account, aimed at preventing the emergence and elimination of software vulnerabilities and containing a list of measures that are recommended for implementation at the relevant stages of the software life cycle.

Various methods for securing the Internet of Things are being actively developed and implemented internationally, including the use of both classic security approaches and new decentralized solutions. Significant attention is being paid to the development of lightweight cryptographic algorithms, secure data exchange protocols, and architectures capable of scaling without compromising security.

Large IoT platforms such as AWSIoTCore, AzureIoTHub, and GoogleCloudIoT use cloud servers that store device-certificate pairs in dedicated hardware secure modules (HSMs) (Isaeva et al., 2025). Devices connect via an encrypted connection using TLS 1.3, and complex cryptographic operations are performed on the hardware. This approach provides a high level of security and full control over keys, but has one major drawback: if a device loses access to the cloud, it will no longer be able to verify the authenticity of its messages. Furthermore, such solutions are too expensive for simple, disposable sensors, especially when protecting thousands or millions of devices.

In addition to the solutions listed above, other solutions are also generating interest. For example, the HyperledgerFabric project offers a modular architecture for creating secure supply chains with

the ability to connect IoT devices. However, it retains the concept of blocks and is resource-intensive to implement. The Helium project uses the LoRaWAN architecture and the Proof-of-Coverage (PoC) model, which enables distributed verification of device presence in the network, also employing elements of a graph structure. Another example is Nano, a cryptocurrency that uses a block-lattice model (a separate blockchain for each account), similar in structure to DAG, and its approach is being applied in micropayment systems between IoT devices.

Research also focuses on the following issues: authentication of devices with minimal computational costs (for example, using blockchain DIDs); protection of confidentiality and integrity of data using hash chains and digital signatures; implementation of a trusted execution environment in microcontrollers to protect against local attacks (Ahmed et al., 2025).

Ultimately, the analysis demonstrates the potential of DAG structures as a compromise between security and resource efficiency. However, there is a lack of methodological recommendations for integrating graph proofs directly into the IoT device stack and for assessing the cost-effectiveness of their deployment outside of cryptocurrency projects. These gaps are the focus of further research. Among the most critical issues are: lack of reliable authentication and encryption; insecure communication channels; the possibility of unauthorized access to devices; vulnerabilities associated with the lack of regular firmware updates; and a centralized architecture that creates single points of failure and problematic scalability.

A literature review confirmed that decentralized models based on directed acyclic graphs can minimize node energy consumption, eliminate mining, and ensure near-instant transaction finality. This is achieved by distributing the verification load among network participants and confirming messages in parallel. This confirms the relevance of developing a proprietary DAG-based IoT security model that can effectively operate under resource-constrained conditions, ensure trust without the involvement of central nodes, and maintain a high level of security as the network scales (Voulgaridis et al., 2025). The identified features and challenges of the information technology object formed the basis for the design solutions discussed below.

It's also worth noting that current approaches to securing IoT devices, based on centralized architectures and classic cryptographic protocols, are insufficiently flexible and unadapted to the specifics of resource-constrained distributed systems. Given this, approaches that ensure security through the data structures and logic of interactions between nodes, without the need for constant access to external trusted authorities, are particularly valuable. The use of a directed acyclic graph (DAG) enables the implementation of such logic, providing a mechanism for distributed confirmation of device actions that is resistant to manipulation and scalable as the network grows. These properties open up opportunities for developing a new level of trust in the IoT environment, in which each participant simultaneously acts as both a consumer and a validator of information.

A System Architecture for Constructing a Directed Acyclic Graph to Ensure the Security of the Internet of Things

A high-level, yet productive and scalable platform is used to create the system prototype. Python was chosen as the programming language, providing: a vast ecosystem of libraries for working with graph structures and web interfaces; easy code maintenance and rapid prototyping; and cross-platform compatibility—the ability to run both on a server and on low-power devices (Alsboui et al., 2025).

The FastAPI framework is used as a REST service: it supports asynchronous request processing, automatic generation of OpenAPI documentation, and is efficient when handling large numbers of concurrent connections. An embedded SQLite DBMS is used to store the graph state and associated transactions; it does not require a separate server and is suitable for test and low-load environments (Wei et al., 2025). If necessary, upgrading to a more powerful engine (PostgreSQL, MySQL) will not affect the rest of the architecture thanks to the storage module's abstraction layer.

Working with DAG objects is accomplished through the specialized NetworkX library, which

simplifies the creation of nodes and edges, while graph visualization in debug mode is facilitated by Matplotlib. The interface for configuring devices and network parameters is designed with easy component replacement in mind: the core modules (DAG and transaction management), presentation layer (API and visualization), device layer, and storage layer are clearly separated and interact via contracts using Pydantic schemas (Kodatsky et al., 2024). Ultimately, the proposed technology stack provides a balance between development speed and architectural flexibility, allowing the system to be subsequently adapted to real-world IoT scenarios without significant modifications.

The designed system is built on a modular approach, ensuring flexible development and easy adaptation to various IoT scenarios. The choice of Python and FastAPI speeds up prototyping and simplifies maintenance, while the default use of SQLite eliminates the need for additional infrastructure during testing.

This preserves the ability to subsequently migrate to more powerful DBMSs and deploy on limited devices without changing the business logic. Clear separation between the DAG management module, storage layer, REST API, simulator, and visualizer ensures component independence and simplifies their testing.The architecture consists of five logically distinct subsystems, each of which performs a distinct role and interacts with the others according to predefined rules. Below, we'll examine the system modules in more detail.

The Graph Management Component module is responsible for selecting nodes without descendants and adding new messages to the graph structure. The processing algorithm selects messages that have not yet received acknowledgements from other nodes and associates the new message with them, forming links in the graph. Each added message receives a unique digital identifier formed from a set of input parameters: the sender's name, the message content, the list of messages to be acknowledged, and a timestamp.

The "Data Storage Component" module is a storage system organized around an embedded database that is initialized upon first launch. It creates tables for devices and messages. Access to the database is handled through an auxiliary module that hides the implementation details and allows for storage to be replaced without rewriting the rest of the code.

The External Programming Interface module connects new devices, transmits messages, retrieves the message history of a specific device, and exports and visualizes the graph structure. Incoming and outgoing data is transmitted in structured text format. The data structures transmitted through the interface are described as strongly typed models, which are used to verify the correctness of requests.

The "Network Simulation" module is designed to test system functionality. It uses a tool that simulates the behavior of multiple devices. Each virtual device periodically generates a message, fetches previous graph nodes, references them, and adds a new entry. By running several such threads, one can simulate real-world load and track how the system responds to an increase in the number of messages.

The "Graph Visual Display" module is used to monitor the system's state. It includes a mechanism for constructing a graphical representation of the current graph structure. Based on the stored connections between messages, a directed graph is constructed, where each message is displayed as a node, and directed edges point to previous messages it acknowledged.

The logic of interaction between components is as follows:

1. When the system starts, the database and necessary tables are automatically created.
2. A new device is added to the registry through the appropriate interface.
3. The device can send a message, and the system selects previous messages for confirmation, generates a new identifier, and stores the information in storage.

4. If necessary, you can get a list of all messages from a specific device, download the full graph structure, or display it graphically.
5. During simulation, the system runs parallel threads, each of which operates as a separate device and sends messages at a specified frequency.

This structure ensures modularity, component independence, scalability, and the ability to replace individual elements without redesigning the entire system. Ultimately, the proposed architecture meets key requirements: decentralized message processing, minimal load on peripheral nodes, and ease of scalability. The described module contracts provide the basis for the prototype implementation, a detailed description of which is provided below.

Software Implementation and Description of the Operation of a System for Constructing a Directed Acyclic Graph to Ensure the Security of the Internet of Things

During the development of a software implementation of a system for constructing a directed acyclic graph to ensure the security of the Internet of Things, a number of key requirements were formulated that determined the architecture and functionality of the future prototype.

As part of the functional requirements, the system must support device registration with the ability to store their identifiers, network addresses, and associated metadata (Cherkesova et al., 2025). Each network node must be able to generate messages with a payload and forward them to a directed acyclic graph for subsequent confirmation. Generated transactions must confirm at least two previous transactions, which are selected in accordance with a specified algorithm. For interaction with devices, a REST API is implemented, including functions for registration, sending messages, retrieving transaction history, and visualizing the graph structure. A separate requirement is the presence of a DAG visualization mechanism, enabling monitoring and analysis of the transaction structure in real time. Furthermore, the system must support the ability to simulate a network of IoT devices, which is necessary for testing its operation under various loads and configurations.

Along with functional requirements, non-functional requirements were also defined to ensure the reliability and practical applicability of the solution. The system must be scalable, meaning it must allow the addition of new nodes without significantly reducing performance. Important requirements include compatibility with typical IoT platforms and the absence of high computing power requirements (Akli, Chougdali, 2025; Kazaryan et al., 2024). The architecture must allow for easy component replacement, particularly of the database management system, without the need to redesign key modules. Resilience to single-point failures is also a priority: the failure of an individual node should not disrupt the operation of the entire network. The implemented algorithms must ensure minimal transaction confirmation times, which is critical for scenarios with limited latency. Finally, the configuration interface must be simple and intuitive, allowing for the rapid configuration of nodes and network parameters without extensive technical expertise.

The system enables interaction between IoT devices through the exchange of messages (transactions) organized into a directed acyclic graph. The process involves several key stages.

	id	name	address
	Fi...	Filter	Filter
1	12	device_1	192.168.0.221
2	13	device 2	192.168.0.5
3	14	device_3	192.168.0.52
4	15	device_4	192.168.0.172
5	16	device_5	192.168.0.222

Figure 1: Registered devices

The first step is registering the device on the network. This occurs via a REST interface, which assigns the device a unique identifier and network address. Device information is stored in a database table. The contents of the registered device database are shown in Figure 1.

After registration, the device generates messages with payloads. Each transaction contains the sender, payload, timestamp, and a list of parent nodes selected by a special algorithm, as shown in the database contents in Figure 2. A unique transaction identifier is generated using the SHA-256 hash function.

	id	sender	payload	parents	timestamp
	Фильтр	Фильтр	Фильтр	Фильтр	Фильтр
1	95d9afbd404c2905e0a1b3d5ad2a95dd948…	device_1	press=24.33		2025-05-28T09:05:07.852531
2	4c0b29cb010c76a4aded064ff496b48d34e…	device_3	temp=46.16		2025-05-28T09:05:07.852531
3	be94d3e4ead1d339a80f89aa8c888f6bc83…	device_4	hum=92.15		2025-05-28T09:05:07.852531
4	ff67d652ed31b58810b819e10c2b968e3c1…	device_5	hum=92.94		2025-05-28T09:05:07.852531
5	c08a0286ac42eed3165bf2aac7e48a7bfcd…	device_2	light=82.45	4c0b29cb010c76a4aded064ff496b48d34e…	2025-05-28T09:05:07.924474

Figure 2: Transaction database

Each new message confirms two previous transactions—this is a property of a directed acyclic graph. The algorithm for selecting vertices with the fewest descendants is implemented in the graph management module. First, all accessible vertices are found, then the number of descendants for each is calculated, and the vertices with the fewest descendants are selected.

IoT devices interact with the system through an application-level software interface. When sending a message, a device accesses a special entry point designed for receiving data. The request contains the name of the sending device, the payload (e.g., sensor readings), and, if necessary, a list of identifiers of existing messages that the new message should acknowledge. If the list of parent messages (messages to be acknowledged) is not specified, the system automatically determines one using a node selection algorithm for a graph without descendants (Kiba et al., 2025). This algorithm searches the storage for all transactions that have not yet been acknowledged by other messages and selects a few, typically two, with minimal load. This allows for even load distribution across the entire graph structure.

The received message is stored in storage, added to the message table, and logically linked to the messages being acknowledged via the "parent" field. This means that the new message becomes a descendant of the messages it acknowledges. The processing sequence for a message from an IoT device is shown in Figure 3.

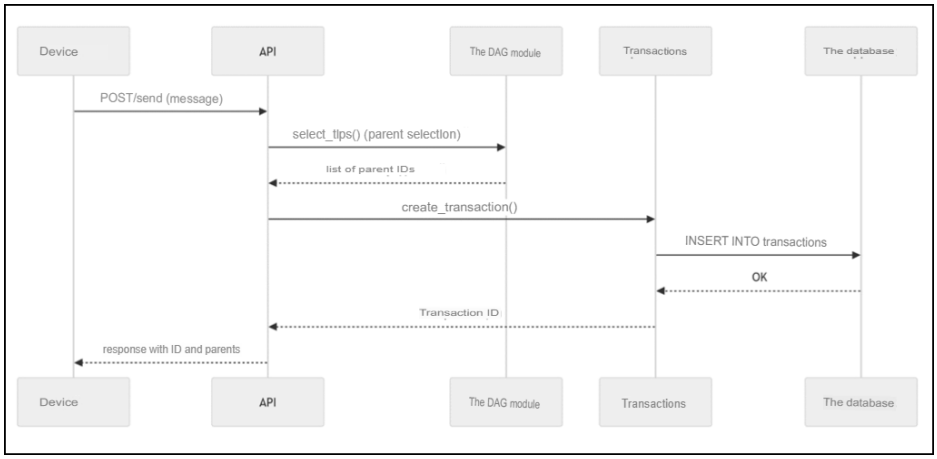

Figure 3: Sequence of processing a message from an IoT device

Thanks to this approach, each new message in the system simultaneously strengthens the reliability of the entire structure. The more nodes confirm the same message, the greater its "weight" and credibility. Thus, forging or deleting an individual message becomes extremely difficult without destroying the entire associated portion of the graph. As a result, the structure of the directed graph grows not sequentially, as in a blockchain, but in parallel, with multiple branches and cross-confirmations. This ensures high resilience to failures and attacks, as well as rapid processing of new messages.

Visualizing a directed acyclic graph (DAG) formed from IoT device transactions is an important tool for monitoring system health, verifying message confirmation logic, and analyzing node load. This functionality is implemented in a separate module designed to extract information about the relationships between messages and construct a graphical representation of them. The goal of the visualization is to display the current DAG structure, where each node corresponds to a single transaction, and directed edges represent "confirmation" links indicating which messages have been confirmed by a given node. An example of the visualized graph is shown in Figure 4.

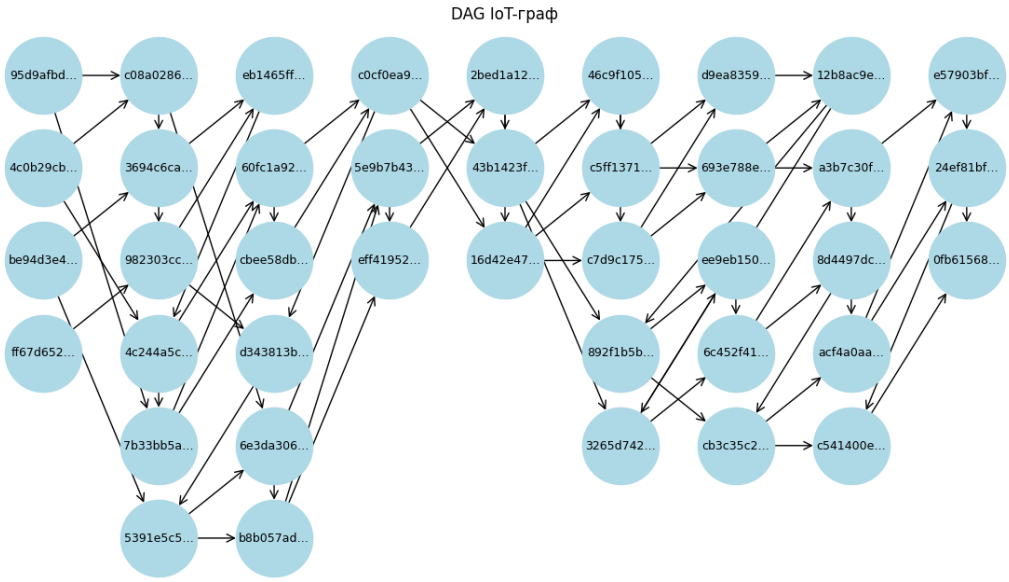

Figure 4: Visualization of a directed acyclic graph

To evaluate the system's performance under conditions similar to real-world use, an IoT device simulator was developed. This tool allows testing the system's behavior under regular message generation from multiple devices, the robustness of the transaction confirmation mechanism, and the uniformity of the directed acyclic graph (DAG) structure. The simulation module is housed in a separate script, structured so that each virtual device goes through a full cycle of actions: Receiving configuration; generating data; selecting parent transactions; forming and sending a new transaction.

Thus, the implemented process ensures continuous and secure interaction between devices through the DAG structure. Next, we discuss the configuration user interface and working with configuration files, which ensure simple and convenient configuration of system parameters.

Description of the Interface and Working with the Configuration

To configure device operating parameters, manage messages, and monitor the DAG structure, the system implements a configuration interface built on an architecture that allows remote access to functions via network requests. The interface ensures consistency between external users and applications and the system core, without requiring manual modification of internal code or the

database. It is implemented through an application-layer protocol that uses standard requests and responses in the form of structured data.

The interface provides the ability to perform basic administrative and operational actions. Operations are organized into separate entry points (methods), with each entry point responsible for a specific function: adding a new device, sending a message, viewing history, obtaining the current DAG structure, or displaying it graphically. The interface's capabilities allow for specific actions:

The user provides unique device characteristics, after which the system registers the device and stores the data in storage. To avoid conflicts, the name and addresses are verified for uniqueness. This operation is necessary to ensure the device can be identified during data transmission. After registration, the device can send a message containing an arbitrary payload. In addition to the payload itself, the sender's name and, if necessary, the identifiers of the messages it acknowledges are specified. If a list of acknowledged messages is not specified, the system automatically determines them based on the current state of the graph.

The interface allows you to obtain a list of all transactions sent by a given device. This is necessary for analyzing activity, reconstructing the chain of events, or debugging the behavior of individual network nodes. While obtaining the graph structure for automated analysis, it is possible to export the current DAG state in structured text format. This feature enables the creation of customized visualizations for external systems, the construction of custom visualizations, link analysis, and the study of confirmation algorithm performance.

To display the graph as an image, the user can request a visual representation of the DAG. In this case, the system generates the image using graphics libraries, returning the image as an illustration. The graph is constructed using information stored in the database; it shows how messages authenticate each other, what relationships they have, and which parts of the graph are generated most quickly.

Working with the configuration interface doesn't require in-depth knowledge of the system's internals. Actions are performed through standard network requests, using familiar data representation formats. This allows for integration with other applications, as well as use within larger platforms or providing control access to remote operators.

As a result, the configuration interface serves not only as an administrative tool but also as an entry point for expanding functionality, connecting new components, and automating tasks (Pankov, Einman, 2022). Its presence makes the system flexible, scalable, and suitable for use both in laboratory settings and within full-fledged IoT infrastructures. Configuration of system operating parameters and device management are primarily accomplished through the application-level software interface. However, basic configuration elements are stored in a local database, the structure of which is defined by a separate SQL script.

An embedded database is used as storage, creating two main tables: one for storing device information, including their names and addresses, and the other for storing transactions, including identifiers, payloads, timestamps, and links to confirmed messages. Access to the database is provided through a dedicated module that abstracts storage operations. This allows the storage mechanism to be replaced with a different one, if necessary, without affecting the rest of the system.

The most common configuration actions include registering new devices, editing existing records, and retrieving information about transmitted messages. Adding a new device is accomplished by accessing the appropriate network method, which writes the data to a table. Changes to parameters can be made manually by accessing the database; however, to maintain consistency and consistency, it is recommended to use the system interface.

Information about the DAG structure and all transactions can also be obtained through the interface. Special commands are provided for this purpose, returning both the complete message history from a specific device and the entire graph. A special tool simulating the operation of several devices is used to test the system's stability. This configuration component allows you to specify the number of

devices, the message generation frequency, and the trigger conditions. It is designed for testing confirmation algorithms and evaluating graph behavior under various conditions.

This configuration approach ensures system flexibility: on the one hand, the user has full control access through a clear interface, on the other, the system's internal parameters are securely stored and processed in a centralized structure without the need to access the source code. This makes the system convenient for both developers and administrators.

CONCLUSION

The development of a software prototype resulted in the implementation of a method for using a directed acyclic graph to ensure resource security in Internet of Things information systems. The developed modules and implemented algorithms fully satisfy the previously outlined requirements. The system provides decentralized device registration and authentication, and also implements the sending of transaction confirmation messages via a DAG mechanism. A SQLite-based data warehouse with a clearly structured table for devices and transactions was created for data storage, and a user-friendly REST API interface using FastAPI was developed for system management and configuration. Visualization of the current graph structure is provided, enabling monitoring and analysis of system operation. Furthermore, the prototype includes a network simulation tool, enabling the verification of system reliability and performance in various scenarios.

The use of a modular approach enabled the development of a system that is easily scalable and adaptable to various hardware platforms and operating conditions. Each module - graph management, API, storage layer, visualization, and simulator - is separate from the others and can evolve autonomously, without impacting the overall architecture. The use of a directed acyclic graph minimized the load on individual nodes, ensured high transaction confirmation speeds, and eliminated the need for a centralized trust node. As a result, the created system can be effectively integrated into real-world IoT networks, increasing their resilience to security threats and reducing the cost of maintaining the security infrastructure.

REFERENCES

Ahmed I, Turki M, Baklouti M, Dammak B. SB-WRW: Balancing fairness and security in IoT-adapted tip selection for the tangle. Internet of Things 2025,34:101819. https://doi.org/10.1016/j.iot.2025.101819

Akapyev VL, Savotchenko SE, Zhukova NA. The development of the Internet of Things and the problems of ensuring its security. Matters Russ Int Law 2024,14:257-67.

Akli A, Chougdali K. Towards tamper-proof trust evaluation of Internet of Things nodes leveraging IOTA ledger. Sensors 2025,25:4697. https://doi.org/10.3390/s25154697

Alsboui T, Al-Aqrabi H, Manasrah A, Artemi M. Toward a secure and scalable IoT: A survey of IOTA-based distributed ledger technologies. Sustain Comput Inform Syst 2025,48:101225. https://doi.org/10.1016/j.suscom.2025.101225

Arunodhai V, Susan LP, Kannimoola JM. EdgeGuard: Real-time rule-based spam detection for IOTA tangle on edge devices. In: 2024 5th International Conference for Emerging Technology (INCET). New York: IEEE; 2024a. p. 1-9. https://doi.org/10.1109/incet61516.2024.10593345

Arunodhai V, Susan LP, Kannimoola JM. EdgeShield: Hybrid real-time attack detection in IOTA tangle via edge devices. In: 2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT). New York: IEEE; 2024b. p. 1-11. https://doi.org/10.1109/icccnt61001.2024.10724609

Baumgartner A, Akkari N, Barua S, Bauschert T. Secure transmission of immutable data for low-power, long-range wireless IoT services. In: 2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). New York: IEEE; 2024. p. 1-6. https://doi.org/10.1109/ICBC59979.2024.10634434

Cherkesova LV, Savelyev VA, Revyakina EA, Polulyakh AR, Sementsov MA. Mechanism for data

recovery as a result of data corruption, infection and/or unauthorized modification. Herald Dagestan State Tech Univ Tech Sci 2025,52:134-46. https://doi.org/10.21822/2073-6185-2025-52-1-134-146

Federal Agency for Technical Regulation and Metrology. GOST R 56939-2024 Information Protection. Secure Software Development. General Requirements. Moscow: FGBU "RST"; 2024.

International Organization for Standardization (ISO), International Electrotechnical Commission (IEC). ISO/IEC 27001 Information security, cybersecurity and privacy protection — Information security management systems — Requirements. 2022a. https://www.iso.org/standard/27001

International Organization for Standardization (ISO), International Electrotechnical Commission (IEC). ISO/IEC 27002 Information security, cybersecurity and privacy protection — Information security controls. 2022b. https://www.iso.org/standard/75652.html

Isaeva OS, Kulyasov NV, Isaev SV. Infrastructure for collecting data and simulating security threats in the internet of things network. Sib Aerospace J 2025,26:8-20. https://doi.org/10.31772/2712-8970-2025-26-1-8-20

Kazaryan MM, Cherkesova LV, Reshetnikova IV, Revyakina EA. Analysis of the possibilities of conducting attacks on the functions of transferring control of the operating system console using active reconnaissance methods. H&ES Res 2024,16:18-29. https://doi.org/10.36724/2409-5419-2024-16-3-18-29

Kiba MR, Zaitseva EV, Kochneva AA, Timokhin MYu. Mathematical model of information distribution in a professional network. Herald Comput Inform Technol 2025,8:27-32. https://doi.org/10.14489/vkit.2025.08.pp.027-032

Kodatsky NM, Revyakina EA, Gazizov AR. System analysis and information processing to solve the problem of detecting breakdowns of computer information storage. Herald Dagestan State Tech Univ Tech Sci 2024,51:87-98. https://doi.org/10.21822/2073-6185-2024-51-4-87-98

Pankov KN, Einman AD. Issledovaniye tekhnologii sistemy raspredelennogo reyestra v sisteme promyshlennogo interneta veshchey s tochki zreniya informatsionnoy bezopasnosti [Research of distributed registry system technology in the industrial Internet of Things system from the point of view of information security]. Sistemy sinkhronizatsii, formirovaniya i obrabotki signalov 2022,13:33-40.

Shamsutdinov RR, Vasiliev VI, Vulfin AM. Intelligent system for monitoring information security of the industrial Internet of Things using artificial immune systems mechanisms. Syst Eng Inform Technol 2024,6:14-31. https://doi.org/10.54708/2658-5014-SIIT-2024-no4-p14

Smirnov NN, Kuznetsov AS. Information support of data processing operations of Internet of things devices in automated information eco-monitoring system. News Kabardino-Balkar Sci Center RAS 2024;26:92-102.

State Duma of the Federal Assembly of the Russian Federation. Federal Law of July 27, 2006, No. 149-FZ "On information, information technologies, and information protection". 2006. http://www.kremlin.ru/acts/bank/24157?ysclid=mnedbgrv6n706840466

State Duma of the Federal Assembly of the Russian Federation. Federal Law of July 26, 2017, No. 187-FZ "On the security of critical information infrastructure of the Russian Federation". Sobranie Zakonodatel'stva Rossiiskoi Federatsii [SZ RF] [Collection of Legislation of the RF] 31.07.2017, No. 31 (Part I), Item 4736.

Voulgaridis K, Karampatzakis D, Sarigiannidis P, Lagkas T. ABS-TD3: Efficient IoT data submission in DAG-based DLTs for digital circular economy. Internet of Things 2025,34:101814. https://doi.org/10.1016/j.iot.2025.101814

Wei Q, Dang S, Ge Z, Li X, Zhang Z. Performance analysis of direct acyclic graph-based ledgers in low-to-high load regime. IEEE Trans Mob Comput 2025,24:12441-55. https://doi.org/10.1109/TMC.2025.3586668

Yang H, Lee S, Kim S. A tip for IOTA privacy: IOTA light node deanonymization via tip selection. In: 2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). New York: IEEE; 2024. p. 494-502. https://doi.org/10.1109/ICBC59979.2024.10634353