Future Generation Computer Systems 140 (2023) 36–52

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

An advanced Internet-of-Drones System with Blockchain for improving quality of service of Search and Rescue: A feasibility study

Tri Nguyen [a,1], Risto Katila [b,1], Tuan Nguyen Gia [b,*,1]

[a] Center for Ubiquitous Computing, University of Oulu, Oulu, Finland
[b] Department of Computing, University of Turku, Turku, Finland

ARTICLE INFO

Article history:
Received 15 April 2022
Received in revised form 18 August 2022
Accepted 1 October 2022
Available online 13 October 2022

Keywords:
Internet-of-Drone (IoD)
Edge computing
Blockchain
Search and Rescue (SAR)
Energy efficiency
Computation offloading

ABSTRACT

Drone-based systems supporting Search and Rescue (SAR) missions help expeditiously improve the possibility of discovering the missing victims. Nonetheless, the current drone-based systems have some limitations, such as the need for humans in control of large-scale drones, drone's energy inefficiency, repercussion of quality of service (QoS) from abnormalities events, shortages of automaticity and real-time interactions, deficiency of swift decisions from artificial intelligence-based SAR, and underestimation of security issues in SAR systems. Therefore, developing a more advanced drone-based system is necessary to overcome these limitations. However, it is challenging to achieve the target due to trade-off relationships of technologies and strict requirements of time-critical SAR. This paper studies the feasibility of an Internet-of-Drones (IoD) system using blockchain and artificial intelligence at the edge to overcome limitations and improve SAR QoS. An advanced IoD system architecture from drones to a back-end system and end-users terminals has been proposed. Furthermore, advanced edge services and artificial intelligence at the edge have been presented for automatically searching for missing persons. In addition, computation offloading approaches have been provided to improve the energy efficiency of drones and reduce system latency. Last but not least, public and private blockchain, including Ethereum and Hyperledger Fabric, for providing secure and decentralized healthcare platform has been investigated and analyzed to enable real-time interaction between healthcare entities and improves healthcare services. The results show that the proposed system helps overcome the limitations and improve the SAR QoS. Besides, the proposed system satisfies the security requirements, including confidentiality, integrity, authentication, authorization, access control, privacy, trust, transparency, availability, automaticity, and tolerance.

1. Introduction

Search and Rescue (SAR) operations function is to locate and aid persons in dangerous situations, e.g., caused by accidents or natural disasters [1]. In a traditional maritime SAR, a rescue team on a helicopter or a rescue boat tries to manually search for victims in the sea areas near the last known position based on Global Positioning System (GPS) or radar signal. However, the SAR performance is not efficient in the case of nighttime, large search areas, and non-GPS. Some of the existing challenges can be surpassed by applying drones and Unmanned Aerial Vehicles (UAVs). Controlling drones is often executed by rescue team members manually controlling to capture videos from above. Depending on the cameras and sensors equipped on the drones, they can be applied in different areas, environments, and times. For instance, drones with high-resolution digital and thermal cameras can search for missing persons in a large sea area in both daytime and nighttime. The drones transmit the videos they capture to a remote controller or control center placed on a SAR boat or a SAR helicopter. Team members have then monitored these videos dedicated to detecting persons in images and videos. Recently, Artificial Intelligence (AI) has been applied to improve the Quality of Service (QoS) of SAR. Particularly, deep-learning models are trained at cloud servers, and the trained models are deployed to computers placed on a SAR boat to detect victims in real-time. The AI approaches aim not to replace SAR team members for monitoring videos but only to assist the rescue team. Notably, the results from the AI approaches and videos can be verified by SAR team members to ensure that all victims are found. Nonetheless, the existing approach of using drones and AI for SAR missions has some limitations, which are as follows:

- The need of SAR team members to control multiple drones for large search areas

* Corresponding author.
E-mail addresses: tri.nguyen@oulu.fi (T. Nguyen), rijukat@utu.fi (R. Katila), tunggi@utu.fi (T.N. Gia).

[1] Authors contributed equally

https://doi.org/10.1016/j.future.2022.10.002

- Energy inefficiency of drones and UAVs
- Repercussion of QoS from abnormalities
- Shortage of automaticity and real-time interactions
- Deficiency of swift decisions from AI-based SAR
- Underestimation of security issues in SAR systems

Due the limitations of the drone-based system for SAR, it is required to develop a more advanced system that can overcome the limitations and maintain high QoS. However, it is difficult to achieve the target as the limitations diverse in different subjects and many of them have trade-off relationships. Utilizing advanced technologies such as swarms of drones [2–4], edge computing [5,6], AI at the edge (edge-AI) [3,7], Internet-of-Things (IoT) [6], cloud computing [8], blockchain [9,10] and cryptographic primitives [11] might provide a solution for the difficulty. Besides, the connectivity between edge computing and blockchain has been utilizing in state-of-the-art research [12,13] due to the close concept that is about the horizontal distribution of computation and storage. These technologies have advantages shown as follows. (i) The swarm of drones enables covering a large search area in a minimal time without manual control. (ii) Edge computing or edge-AI brings cloud paradigms, AI, advanced services closer to data generators. In addition, edge computing addresses unfit fundamentals in the cloud computing paradigm. For example, edge location, low latency, location awareness, geographical distribution are some of the fundamental characteristics of edge computing [6]. (iii) IoT enables real-time streaming and interactions with remote experts and facilities; meanwhile, cloud computing facilitates complex computation, AI and big data management. (iv) Blockchain and cryptographic primitives address security issues such as confidentiality, authorization, integrity or access control. Nonetheless, it is challenging to efficiently and harmoniously apply all these technologies into a system while fulfilling all the requirements such as low latency, energy efficiency, high level of accuracy and security.

In this paper, we study feasibility of the advanced drone-system using the aforementioned technologies for SAR missions. First, we design an advanced Internet-of-Drones (IoD) system architecture from a swarm of drones to back-end system for real-time streaming and interactions with remote experts. Then, several computation offloading types are presented and investigated to find out the suitable approach that helps achieve a high level of energy efficiency at swarm of drones. To demonstrate edge-AI and achieve the fair analysis of computation offloading, different YOLO versions for real-time object/human detection are implemented at edge devices [14,15]. Latency and energy consumption results in case of computation offloading are analyzed. In addition, security requirements for the IoD systems for SAR missions are presented and used for analyzing the security satisfaction of the proposed system. Furthermore, the performance of cryptographic primitives and blockchain, including Ethereum and Hyperledger are comprehensively analyzed. Smart contracts (e.g., for getting assistance from hospitals) are designed and tested. Moreover, edge services for maintaining QoS especially in case of abnormalities such as malfunctioned drones are presented. Last but not least, a complete system from a swarm of drones, edge servers, blockchain, cloud servers to end-user terminal is built and used for experimenting.

The structure of the paper is as follows: Section 2 provides related work that presents state-of-the-art computation offloading and blockchain-based drones. Section 3 discusses preliminaries, including the background of blockchain technology and security requirements for drone-based SAR systems. Next, Section 4 shows the proposed system's architecture, while Section 5 shows different computation offloading types. Section 6 presents the system setup and implementation, and the experiment results are in Section 7. Finally, Section 8 indicates future work and conclusion.

2. Related work

This section presents the state-of-the-art approaches for drone-related computational offloading and blockchain-based approaches for drone-based applications.

2.1. Computation offloading in drone-based systems

Hou et al. [16] present a fog-based computation offloading approach for a swarm of drones. A swarm can be divided into groups in which each group consists of small drones that can be considered fog computing nodes that can intercommunicate and share tasks. The proposed approach uses a heuristic algorithm to improve energy efficiency and reliability, and reduce latency. The simulated result shows that the proposed approach achieves the target efficiently. The proposed LRGA-MIE algorithm improved energy efficiency by around 41% to 45% when compared to other algorithms with 0.5 MB input data size.

Jo et al. [17] introduce a selective computation offloading approach for a drone-based system to improve energy efficiency and optimize completion time. The approach uses a cost analysis model to assist task offloading decision-making. The approach is evaluated by simulating offloading exclusively to a cloud server. Tasks are either executed locally in drones or offloaded to a cloud server when appropriate. Results show that in the effect of instructions length, energy consumption does not increase in offloading tasks, but executing tasks in drones increases it linearly.

Studies [18,19] present a control and computation offloading framework using Function-Centric Computing (FCC). A target application is divided into different standalone microservices and can be processed at different computing nodes. The frameworks facilitate decision-making for computation offloading to edge or cloud and optionally using FCC. The presented frameworks were tested using a GENI testbed. In [19], results show that when the cloud resources are limited, FCC can reduce latency and have a smaller cost compared to only using edge computing.

In [20], a computation offloading approach between a clusters of drones was proposed. In the approach, each cluster will have a leader which queries adjacent clusters' available resources for task offloading. The decision is made by the cluster leader which estimates the response time and processing-related results via computing power, task size, bandwidth, and data rate when resources are available. The approach can increase the drone's operational time and reduce latency.

Callegaro et al. [21] present a computation offloading approach for edge-assisted UAV systems. In the approach, a task is locally processed at a drone or offloaded to an edge-based server depending on processing latency and energy consumption. The approach uses offloading tasks based on the Markov Decision Process formulation algorithm. Results show latency reduction in the proposed offloading scheme.

Lin et al. [22] propose an energy-efficient computation offloading approach for UAV-assisted mobile edge computing. The authors designed a centralized algorithm for Nash Equilibrium extraction, the strategy profile that maintains social benefits and does not breach individual rationality. The simulation results show that the proposed approach achieves good performance and helps save the energy consumption of UAVs.

In [23], the authors present an energy-efficient design of computation offloading for drone-based systems. The authors aim to achieve a high level of energy efficiency in drones while maintaining a high level of services, especially in case of abnormalities such as natural disasters causing damage to communication systems. A successive convex approximation-based alternating algorithm is designed to maximize the energy efficiency of a

Table 1
Comparison of previous works related to computational offloading from drones.

Pub	Val	Advantages	Limitations
[16]	S	Improved latency, reliability, and energy consumption	Latency model does not consider computational cost of executing LRGA-MIE algorithm or downlink transmission of task results
[17]	S	Selective offloading to edge server improves latency and energy efficiency in drones	Omits individual drones' available processing (handled in clusters) or bandwidth capability
[18]	S	Dynamic offloading with FCC approach. Effectively uses available resources in drones, ground stations, and cloud	Does not consider drones energy consumption as a parameter but only QoE and latency
[19]	E	Adapts to occlusions. Realistic geo-distibuted edge/core cloud testbed	Energy consumption in drones is not evaluated and communication bandwidth is disregarded offloading decisions
[20]	T	Effective task distributing and computation offloading at the edge	Considers only offloading from a drone cluster to another. Purely mathematically modeled
[21]	S	Probing the network and edge server to make fast decisions about offloading	Transmission probability model does not take in account link quality that reduce the successful transmission rate
[22]	S	Achieves good performance and lowers energy consumption	Excludes task balancing between drones or other edge devices
[23]	S	Operating even when there are occlusions achieving good energy efficiency and latency	Connection reliability and latency is not considered
[24]	E	Practical experiments showing that bandwidth, energy efficiency and latency are improved	Does not cover distributing and computing tasks partially in the edge or cloud in a hybrid strategy
[4]	E	Blockchain as part of system to improve security	No actual experimental implementation of computation offloading when running human detection applications

T = Theoretical, E = Experimental, S = Simulated, Pub = Publication, Val = Validation.

drone. Parameters such as battery level, data causality, and drone speed are the principal measurement metrics in the proposed approach that improves the energy efficiency of the UAVs by additionally using offloading.

In [24], the authors introduce a drone-based system architecture of computation offloading to improve latency, bandwidth, and energy efficiency. The system architecture consists of drones, edge-based devices, in-cloud deep neural networks, and control station dashboards. The authors conduct practical experiments to evaluate the proposed architecture by comparing cloud-based computation offloading and edge-based processing regarding energy consumption, bandwidth, and latency. Results prove that offloading heavy computational tasks such as the computer vision from UAVs in the article helps to reduce energy consumption.

Although different computation offloading approaches in previous works have shown to improve the energy efficiency of drones and maintain low latency, they have many assumptions and have not investigated the topic extensively. The comparison is presented in Table 1. Many of the presented approaches are simulated and simplified cases of local processing at a drone or offloading to a fog node or the cloud. Often there is very little or no consideration for data security.

In our previous work [4], IoD and blockchain system architecture for SAR was researched. Computation offloading was briefly presented to show the architecture capable of facilitating edge services that improve the QoS. Hyperledger integration was introduced to provide trust, security, transparency, and decentralization. However, computation offloading and blockchain were not investigated in-depth, analyzed, or fully implemented. This article extends the previous work by presenting a complete IoD system with edge computing and blockchain for SAR. Particularly, computation offloading approaches have been presented and implemented in edge devices used in actual drones. Furthermore, computation offloading has experimented with real-time human detection tasks tested with video images captured with a drone. In addition, Hyperledger and Ethereum have been implemented and compared. The experimental results were comprehensively investigated and analyzed.

2.2. Blockchain-based Internet-of-Drone systems

Lv et al. [25] propose adopting blockchain technology to solve privacy issues in UAV big data. This consideration is to reduce the computation cost compared to existing approaches by providing theoretical significance to protect the privacy of UAV big data. Furthermore, the blockchain concept is utilized to prevent data modification and track transactions, along with the NTRU for encryption and decryption. Like this idea, [26] points to the usage of blockchain to handle privacy issues in 5G-enabled drone communication. Wu et al. [26] indicate the role of blockchain in identity management, avoidance of fake drones, and resistance to malicious attacks on drone communication through blockchain characteristics. For example, blockchain-based solutions include consensus of drone networks, data privacy protection based on cryptographic schemes, drone identity management, and access control.

Studies [27,28] propose integrating those technologies based on the interest of blockchain in communication and AI. Gumaei et al. [28] utilize a deep neural network using radio signals to identify drones and detect flight modes. The system leverages blockchain as the middle layer of connection between drones and edge devices to gain the secure transmission of drone identification and flight modes. On the other hand, [27] utilizes a consortium blockchain as a public database or secure data management, where the data is retrievable in the COVID-19 pandemic. However, these works detail a cryptographic scheme for a blockchain-based system and supervisor learning evaluation instead of a particular architecture for the system.

Another study [29] mentions the prominent utilization of blockchain to solve the challenges of drone networks. In detail, the drone layer is the connection among drones for specific operations as miners for blockchain, whereas the resource layer is about blockchain setup with resource allocation. An edge service provider sets the management layer to manage resources and decisions, for example, offloading computation from drones. Nevertheless, this work is based on a permissionless blockchain

Table 2
Summary blockchain-enabled IoD.

Pub	Type	Application	Weakness	Lack of security requirements
[25]	-	UAV big data privacy protection	Focusing on homomorphic encryption of NTRU and the lack of information related to blockchain technology design	Authorization and automaticity
[26]	Public	Privacy-preserving 5G-enabled drone communication	Lack of detailed experiments in the design	Privacy issue of drones and users information
[27]	Private	Secure data management in AI-empowered pandemic	A complicated system with many light chains and the main chain requires powerful drones	Privacy issues and authorization
[28]	-	5G-enabled drone identification and flight mode detection	Required powerful drones and lack of experiments in blockchain design	Privacy challenges, automaticity, and authorization
[29]	Public	Drones to manage a public blockchain	Required powerful drones for blockchain maintenance	Automaticity, privacy, authorization, and authentication
[30]	Public	Secure authentication model	The use of public blockchain reduces the privacy and performance	Automaticity, authorization, and privacy
[31]	Private	Dynamical cache data for offloading	Lack of experiments in communication and encryption complexity	Privacy and tolerance issues

Pub = Publication.

requiring that a drone contributes to the blockchain through a consensus-based data storage at the edge layer. Thus, the power computation of drones is the need for this work, which leads to an issue in energy consumption.

Consideration for an authentication based on blockchain is from a study [30]. Yazdinejad et al. [30] mention a suitable architecture for a blockchain-based drone in a zone-based network of a smart city scenario where each drone controller represents a specific zone and maintains the blockchain. Due to a public blockchain, the blockchain architecture design is based on a Delegated Proof-of-Stake consensus to gain better performance than a traditional Proof-of-Work (PoW) consensus. However, despite the benefits of the public blockchain, the permissioned blockchain provides better performance with the identity of drone controllers.

Commonality [31] to our proposal is about leveraging the blockchain in a drone network via the support of mobile edge computing. Following [31], the architecture consists of three main layers: wireless sensor networks, drones, and MEC servers. The drones select the closest MEC server for offloading tasks from IoT devices at wireless sensor networks via a smart contract as the policy of offload tasks. Further, this work points to the offloading policy deployed with smart contracts at MEC servers. Hence, this drone architecture cannot provide a high flexibility level as a hierarchical setup of IoD. Also, the offloading policy can be deployed at different drone layers instead of the only deployment of smart contracts at MEC servers.

Although the state-of-the-art blockchain approaches for IoD provide some levels of security such as trust, transparency, availability, and tolerance, they still have weaknesses shown in Table 2.

3. Preliminaries

This section describes the preliminaries of blockchain technology and security requirements for SAR.

3.1. Blockchain technology

The success of blockchain is from the construction of consistency in the shared database among participants in a decentralized environment. Consistency, a primary feature in a decentralized system, provides a unique state among participants. However, to reach consistency and gain decentralization, the system requires reliable communications and agreements to update

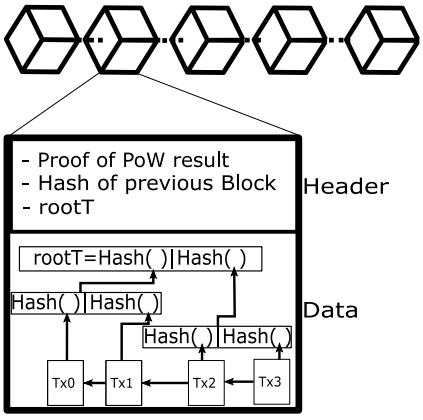

Fig. 1. The blockchain structure.

the system's state through a well-designed structure integrating a consensus mechanism and cryptographic schemes as a blockchain model.

The formation of blockchain encompasses data blocks connecting to form a chain of data blocks based on the hash pointer concept. Following Fig. 1, each data block wraps a chronological order of transactions in the data part, while the header information of a data block contains the result of the Merkle tree [32] and the hash value of the previous block. Depending on the consensus mechanism to form the agreements among participants, each block stores other information that proves the correctness of block formation. For example, in Fig. 1, the PoW result is that a participant resolves a hashing puzzle at a consensus round. With this result, the participant has permission to wrap up a block candidate, which is then broadcast to other participants for evaluation, whether confirmation or not. By constructing the blockchain concept, a blockchain-based system detects late modification from any faulty participants who aim to break the system's consistency.

Blockchain technology is a fit solution to form and gain advantages from the system's decentralization. As the prime benefit of decentralization, fault tolerance can be reached with numerous participants contributing to the system. Furthermore, a blockchain-based system gains trustworthiness due to the immutability and transparency of the system's data. With the formation of smart contracts called Ethereum [33,34], the system

can deploy decentralized applications or even decentralized autonomous organizations. For example, once a drone reaches a low battery notification, the system requests another drone nearby to replace it. As another example, the system ordinarily operates with the remaining participants with the crash of a participant maintaining the blockchain.

3.2. Security requirements for the drone-based SAR system

R1: **Confidentiality**: Information such as administrative data, data exchanged between rescue team members, or specialists' instructions must be confidential. Only authorized persons who are responsible for the tasks can access the information. When IoD systems are used for SAR missions, data confidentiality needs to be considered as transmission happens over the Internet. Every part of the IoD system needs to be secured to guarantee a high level of confidentially. A combination of different secure approaches such as cryptography algorithms, cryptography primitives, and blockchain-based approaches can be a solution. For example, cryptography primitives (e.g., AES-128, AES-256, or ChaCha20Poly1305) can be applied to small drones and big drones as they ensure some levels of security while they do not require heavy computation. In contrast, edge servers that are much more powerful than drones can work as blockchain nodes of the blockchain network.

R2: **Integrity**: In IoD, data transmitted over the wireless networks (from drones to other drones or from drones to control stations) must be reliable, complete, and consistent. The transmitted data must be protected and has not been modified by a third party. It is necessary to apply encryption and some validation or error checking methods to ensure that sensitive data is safely transmitted and correctly stored. For example, cryptography algorithms and primitives (e.g., AES-128, AES-256, or ChaCha20Poly1305) can be applied.

R3: **Authentication**: When users such as specialists and administrators want to access the IoD system, their authentication needs to be checked and verified. The system can use different methods (e.g., a password, a security token, or facial recognition) to verify the user. Correspondingly, users can use a mobile or web-based app to provide the required information.

R4: **Authorization and Access Control**: The system needs to categorize group users based on their authentication and responsibility. Group users have different rights to access specific data.

R5: **Privacy:** Personal information and data related to victims, rescue teams, or specialists must be legitimately handled. When the data is used for any purpose including health operations and activities, it must be passed through victims' consent. In addition, it is required that any misuse of data must be avoided. The system must respect and satisfy General Data Protection Regulation (GDPR).

R6: **Trust**: Persons participating in SAR missions and parties joining the BC healthcare network need to feel safe when using the IoD system and BC healthcare network. In other words, the system and its programs/applications must be trusted when being used. There must be no secretly executing harmful or unauthorized programs. In addition, the system needs to have mechanisms to prevent users from executing programs undesirably. Last but not least, the system needs to categorize the information into levels such as unclassified, confidential, secret, and beyond. These levels correspond to users' authorization levels.

R7: **Transparency**: Observing information about collaborations among organizations in the SAR system depends on transparency. This feature supports the victims' situation to other responsible organizations. As an example of the consideration of data transparency, the rescue plan can provide specialists or victims' families with more help or observers.

R8: **Availability**: Information related to SAR and victims should always be available when needed. For example, medical doctors' information related to first aid treatment steps when finding a victim must be available for doctors who will decide on further treatments to save the victim.

R9: **Automaticity**: Due to the importance of swift reactions in SAR, automaticity assists the formation of connectivity in organizations. For instance, the rescuer can automatically contact extra help, such as medical suppliers or hospital specialists.

R10: **Tolerance**: Tolerance is one of the most crucial features of a SAR system. For example, during a search and rescue progress, the victims face a dangerous situation if there are some crashes from communication or computation in the system. Therefore, a SAR system tolerates issues from faults.

4. System architecture

The proposed system architecture shown in Fig. 2 has four main layers: a small drone layer, a big drone layer, an edge server layer, and a blockchain network and application layer.

4.1. Small drone layer

The small drone layer consists of drones which are resource-constrained devices. They have low battery capacity, low memory, and low capability for processing and storing. In this paper, these drones are named small drones and can be equipped with different types of cameras and sensors. For example, GPS sensors, RGB, and thermal cameras can be used to enable daytime and nighttime monitoring. In some cases, such as autonomous driving, other sensors such as Lidar sensors can be applied. Lidar is less affected by visibility, lighting, and noise but is often heavy and costly. When more sensors and cameras are used, the flying time of the drones reduces as the load increases. Therefore, selecting proper sensors and cameras is necessary. For example, the RGB cameras for small drones often support 10 MP–48 MP images and 2K–8K video with 30/60 Frame Per Second (fps). The cameras' weight is usually less than 600–700 g.

A small drone often has a small computer board with limited resources such as low memory and computational capability. For instance, embedded boards such as Rasberry Pi 4 or Jetson Nano can be used because they are lightweight, low-power, and provide decent computing capabilities such as data compression and lightweight image processing. A small drone is often equipped with a Wi-Fi module as Wi-Fi supports high data rate transmission satisfying the requirements of high-resolution videos. For instance, captured videos by a small drone are sent via Wi-Fi to a big drone. For achieving some security levels, the drones use solid passwords and Wi-Fi Protected Access 3 (WPA3) when accessing/joining the Wi-Fi network. The password can be regularly updated after each SAR mission. Additionally, cryptography primitives such as AES-256 and ChaCha20 can be applied to encrypt the data before being sent over the Wi-Fi network.

The small drone can fly at different heights (e.g., from 20 m to a few hundred meters) depending on the drone's specifications and the environment. Flying a drone at a higher altitude enables covering a larger area in one go, but the drone may meet stronger wind and has low signal strengths. Based on our previous experiments with customized small drones, a height of 50–100 m can be a decent option. At this altitude range, Wi-Fi signal strength is adequate, and objects in captured images are detected reliably. Depending on the situation and environment, a shorter height may be more suitable to increase the visibility of objects/humans in the captured videos. In the proposed architecture, several small drones (e.g., 4–8 small drones) together with a big drone form a cluster in which a big drone is a cluster head. The number of small

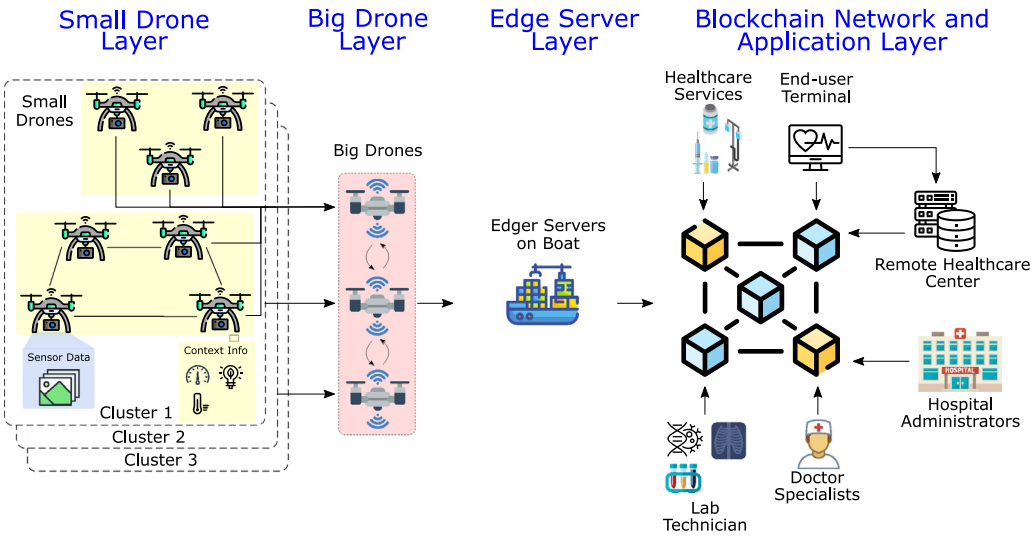

Fig. 2. The IoD architecture with the integration of edge computing and blockchain.

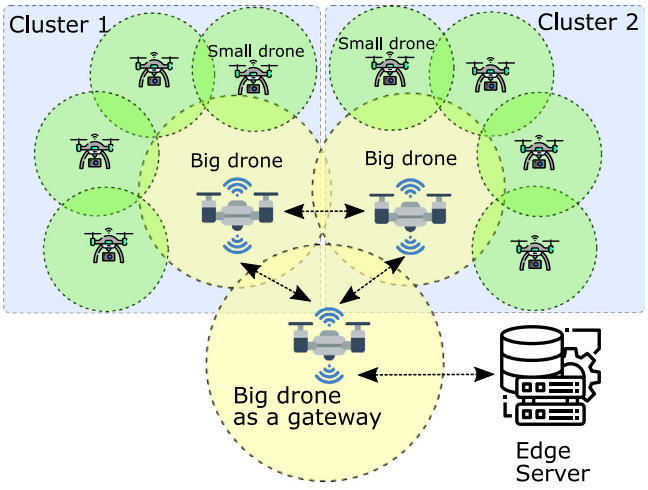

Fig. 3. The cluster of drones and the communication of the drones in the proposed architecture.

drones can be flexibly set by a rescue team depending on SAR missions. Increasing or reducing the number of small drones in a cluster does not affect the working activities of cluster members.

Each cluster covers a particular area in which each small drone covers a part of the area. When the small drones are launched, based on the flying plan, the small drones automatically fly from a control center to the search area where they cover, as seen in Fig. 3. When the small drones reach their responsible areas, they can fly circularly in clockwise manner. The radius of the flying circular is set based on the drone's camera view, flying height, and drone types. Depending on the SAR missions, a rescue team can set other flying modes such as polygon, grid, double grid, or free flight. When the search area is large, several clusters can be used shown in Fig. 3.

4.2. Big drone layer

The big drone layer consists of big drones that can carry more loads (e.g., a 5–6 kg load) and have more powerful resources than small drones. For example, a big drone can be equipped with high-resolution RGB and thermal cameras with lenses for zooming and working in both daytime and nighttime. A big drone is also equipped with powerful embedded boards that can run complex algorithms. Particularly, embedded boards can have multiple Central Processing Unit (CPU) and Graphics Processing Unit (GPU) cores (e.g., NVIDIA Jetson Xavier NX with 384 GPU cores or NVIDIA Jetson AGX XAVIER with 512 GPU cores). These embedded boards can facilitate edge-AI and advanced edge services. Nonetheless, big drones are larger, heavier, and more expensive than small drones. For example, a small drone can be around 2–3k Euros, while a big drone can be 10–18k Euros, as shown in Table 4. Similar to small drones, big drones also use Wi-Fi for communicating with small drones and edge servers placed on a SAR boat. Strong passwords, WPA3, cryptography algorithms,

Table 3
Symbol descriptions.

Symbol	Meaning
L_c	Latency comparison (s)
E_c	Energy comparison (J)
L_{ps}	Processing latency at small drone (s)
L_{Pb}	Processing latency at big drone (s)
$L_{Tresult}$	Latency of result transmission from a small drone to a big drone (s)
L_{Pb}	Processing latency at big drone (s)
L_{Tdata}	Latency of raw data transmission from a small drone to a big drone (s)
E_{ps}	Energy of data processing at small drone (J)
E_{pb}	Energy of data processing at big drone (J)
$E_{Tresult}$	Energy of transmitting the processed data from a small drone to a big drone (J)
$E_{Tresultb}$	Energy of transmitting the processed data from a big drone to an edge server (J)
E_{Tdata}	Energy of transmitting raw data from a small drone to a big drone (J)
E_{Tdatab}	Energy of transmitting raw data from a big drone to an edge server (J)
V_s	Voltage supply for small drone's board (V)
V_b	Voltage supply for big drone's board
C_s	Battery capacity of a small drone (Ah)
C_b	Battery capacity of a big drone (Ah)
B_s	Battery level of a small drone (ranging [0%, 100%])
B_b	Battery level of a big drone (ranging [0%, 100%])
R_b	Ratio level (ranging [0,1])
E_{tis}	Energy of transmitting images from a small drone to a big drone (J)

Table 4
Drone specifications.

Type	Main components	Price (€)
Small drone		1.2k–1.4k
	Holybro x500 frame and propellers	100
	4 Holybro ESC & 4 880 kV motors	200
	Flight controller Pixhawk 4	220
	Lidar ranging sensor TFmini	50
	RGB Camera	200
	GPS Pixhawk 4 + Wi-Fi Dlink + Teleradio	100
	Basgibg 5500 mAh, 14.8 V battery	70
	Operating board(s)	
	Raspberry Pi 4 B	90
	Intel UP Squared board	250
	Nvidia Jetson Nano	120
	Other components	100
Big drone		9.1k–9.7k
	DJI S1000 drone	2k
	DJI FLIR Zenmuse XT2 (Thermal + visual camera)	6k
	Operating board(s)	
	Nvidia Jetson NX Xavier	400
	Nvidia Jetson AGX Xavier	1k
	Tattu 26000 mAh Lipo Battery Pack	600
	Other components	100

and primitives can be applied to achieve some security levels. In addition, a big drone uses a whitelist to store the MAC addresses of the authorized drones. Only drones on the whitelist can access a Wi-Fi network and communicate with the big drone. A big drone can regularly scan other nearby drones using Wi-Fi to reduce the risk caused by unauthorized drones. When it detects an unauthorized device that is not registered in the whitelist, the big drone can send an alarm to an edge server. If Wi-Fi jamming occurs, the system may find the source (e.g., unauthorized Wi-Fi devices), causing the jamming easier.

A big drone can have one or several roles defined by the situation. Particularly, when a big drone does not have any camera, it can act as an edge device for providing edge services such as distributed data processing and compression. However, when a big drone is equipped with cameras, it has a monitoring role. Similar to small drones, a big drone searches for persons in its cover area automatically based on the flying plan. In addition, a big drone can act as a verifier. Particularly, when the system/edge servers detect a person from a captured video, it can send a verifying instruction to the big drone of the cluster that captures videos having detected persons. In this case, the big drone moves closer to the area where the person is detected, or a big drone can zoom into the area. Then a new video is captured and transmitted from the big drone to an edge server to detect the person once more. It is noted that the big drone has a higher resolution camera than a small drone. Therefore, the detection result from the higher resolution video might be more reliable. Due to the importance of the SAR mission, it is necessary to double-check the results.

A big drone constantly monitors the input sent from small drones. When the big drone cannot receive data from a specific small drone for a period, the big drone sends a command to the small drone. If there is no response from the small drone after a predefined period (e.g., a few minutes), the big drone will send an alarm and a request to an edge server placed on the SAR boat or SAR helicopter. After receiving the request, the edge server/control center can send another small drone to the area where there is malfunctioned small drone. This mechanism helps prevent missing search areas and detect malfunctioned drones. At this time, the whitelist is updated with a new small drone. Similarly, big drones are continuously monitored by edge servers. If there is no response from a big drone for a period, the edge server will send another big drone to investigate the case and replace the malfunctioned big drone to continue the tasks.

4.3. Edge server layer

The edge server layer consists of powerful edge servers placed on a SAR boat or helicopter. The edge servers are supplied with power from an electrical socket and have large storage capacity, memory, and computational capabilities. Therefore, edge servers can run different complex algorithms and applications, including deep-learning human detection algorithms. In addition, an edge server facilitates many advanced services, such as distributed data storage, data processing, data compression, push notification, and edge-AI, which were discussed in detail in our previous papers [6, 35,36]. In distributed storage, edge servers can be equipped with different databases to store the categorized data (e.g., unclassified data or secret data). Depending on the authorization level, different databases can be accessed.

The server connects with a big drone via Wi-Fi and connects with a cloud server via 4G/5G. In another view, edge servers are the bridge between the internal networks of drones and external networks of outsiders such as a hospital or remote medical experts. Correspondingly, blockchain does not affect the system's drones in terms of energy consumption and latency. Last but not least, edge servers are nodes of a blockchain network as they can satisfy the requirements of heavy computation and large energy consumption. In other words, edge servers synchronize and maintain a unique blockchain as the public database for the whole system. With this perspective, blockchain data supports the shred of evidence for audit and analysis of danger areas. Therefore, rescue teams can receive optimized tactics from the system based on analyzed data.

4.4. Blockchain network and application layer

The blockchain network and application layer includes cloud servers and services such as global data storage, push notification, data management and analytics, machine learning, and AI. Commercial cloud servers and services offered by Google, Amazon, or Microsoft are available for utilization. They offer many advantages and satisfy the requirements of the SAR missions in terms of security. For example, large amounts of data, including a history of data, can be correctly stored for long periods. In addition, CPU, GPU, and memory resources can be scaled based on demand with low costs. This layer also consists of the terminal application that can access real-time data and provide input for the system, including instruction from an expert. The application layer can be interfaced through a web browser or a mobile application. Accessing data requires an end-user to provide his/her authentication, for example, logging in with a username and password or logging in with face detection.

The formation of blockchain requires numerous communications for synchronization, powerful computation for consensus validation and cryptographic schemes, and even storage. For example, with Bitcoin PoW, the consensus process demands as much as powerful computation to seek permission for a new block formation. Also, for verification, the blockchain participants need a whole history of transactions by which the current transactions demonstrate the correctness of the system's current state. Therefore, the proposal utilizes powerful servers as edge servers for blockchain deployment.

The primary usage of the blockchain layer is to gain consensus and avoid conflict in data storage for the system. Due to the power of computation demand in blockchain maintenance, the edge servers in our proposal handle blockchain construction

through communication and consensus processes. A blockchain participant, thus, gathers big drones' data and warps it into a block candidate for blockchain extension. However, the block candidate needs to achieve agreements from other participants before attaching to the current blockchain by proving the correctness without conflicting with previous confirmed data. Therefore, blockchain requires a suitable consensus mechanism for the use case, and especially in this scenario, it is desirable to use Raft [37] in Hyperledger Fabric as an example of a permissioned blockchain. Raft is a leader-based consensus that selects a leader per consensus round to form a block candidate through an election algorithm. Meanwhile, the PoW is a consensus as Proof-of-X consensus [38] in the usage of Ethereum, a permissionless blockchain.

Besides, the blockchain data can be various information and system operations with smart contracts to gain autonomous processes. For example, participants' information, transaction execution, smart contract agreements, drones' multimedia data, and drones' information as power, metadata, and location are possible information deployed in the blockchain. Therefore, the system expects to swiftly and autonomously shape recommendation and connectivity among entities with efficiency through the platform with diverse entities' information.

5. Computation offloading

Computation offloading from the standpoint of IoD systems can be described as switching computational tasks from drones with limited resources such as low-speed CPU and GPU, low memory and battery capacity to more powerful drones or a control station that is capable of processing heavy algorithms in low latency. Computation offloading aims to extend the flying time of drones while satisfying the time requirements. In the proposed system shown in Section 4, computation offloading can be categorized into four types.

- Type 1: From a small drone to a big drone
- Type 2: From a big drone to an edge server
- Type 3: Between big drones
- Type 4: From an edge server to the cloud server

In many general cases, the third and the fourth types are often not focused on because of their limitations. Notably, each big drone is responsible for collecting or processing data from its cluster drone members. Correspondingly, big drones often do not have available resources. Sometimes it might not be beneficial to perform the computation offloading between big drones. However, computation offloading between big drones is still needed in some exceptions. For example, when a connection between a big drone and an edge server is disconnected due to environment or noise, some computational tasks need to switch to an adjacent big drone. On many occasions, the fourth computation offloading type is not applicable due to the long latency between an edge server and a cloud server. In these cases, processing data locally at an edge server is more efficient considering latency. The edge server is powerful enough to process heavy computational tasks. Hence, this paper focuses on the first and second computation offloading types that help extend the operating time of drones while not increasing the system latency.

5.1. Computation offloading type 1

Computation offloading decisions can be made based on either latency, energy efficiency, or both. Latency-based decision-making is shown in formula (1):

$$L_c = (L_{ps} + L_{Tresult}) - (L_{Tdata} + L_{Pb}) \begin{cases} \text{offload}, & L_c > 0 \\ \text{local process}, & L_c \leq 0 \end{cases} \quad (1)$$

In the case of targeting latency reduction, the total latency of data processing and transmitting is examined first by calculating latency for processing at a small drone. Then, calculate the total latency of transmitting data from a small drone to a big drone and do data processing there. These latencies are compared for deciding on computation offloading task-by-task basis. The lower latency approach is chosen to reduce the total latency of the system and ensure maximum efficiency. Energy-based decision-making is shown in the formula (2):

$$E_c = E_{ps} + E_{Tresult} - E_{Tdata}$$
$$\begin{cases} \text{offload}, & E_c > 0 \text{ and } \frac{E_c}{3600 \times V_s \times B_s \times C_s} \geq R_b \times \frac{E_{pb}}{3600 \times V_b \times B_b \times C_b} \\ \text{local process}, & E_c > 0 \text{ and } \frac{E_c}{3600 \times V_s \times B_s \times C_s} < R_b \times \frac{E_{pb}}{3600 \times V_b \times B_b \times C_b} \\ \text{local process}, & E_c \leq 0 \end{cases} \quad (2)$$

In case energy efficiency for a small drone is focused, a total energy consumption (e.g., called E_{total}) consisting of energy consumption of data processing at a small drone and energy consumption of transmitting the processed result is firstly calculated. The total energy consumption is compared with the energy consumption of transmitting raw data (i.e., unprocessed images). If the total energy consumption E_{total} is larger and the energy-saving level at a small drone is higher than the energy spending level at a big drone, computation offloading from a small drone to a big drone is carried out. System administrators note that the ratio level is pre-defined depending on the application and drones' specifications. In general, the ratio level is set to 1 for most cases as the operating times of a big drone and a small drone are similar. In case the operating time of a big drone is much larger than a small drone, the ratio level can be set to less than one based on the different operating times. When the total energy consumption E_{total} is larger than the energy consumption of transmitting data, but the energy-saving level at a small drone is not larger than the energy spending level at a big drone, local processing is made. Finally, if the total energy consumption E_{total} is less than the energy consumption of transmitting raw data, local processing is also carried out.

The energy consumption of an embedded board when performing a task depends on the latency spent executing it. Therefore, when processing latency reduces, energy is saved if the processing power consumption does not increase in the same proportion. In our case, the energy consumption of an embedded board of a small drone includes energy consumption of data processing and energy consumption of data transmission, which relies on the power of data processing and transmission power, respectively. These power consumptions are different, and the different level depends on the embedded board and a communication protocol. Therefore, it does not always guarantee that the highest reduced latency can help save the most energy. When computation offloading decision is based on latency and energy consumption, one of them should have a higher priority. Although latency is crucial for the SAR systems, an increase of a second or two seconds almost does not significantly impact the QoS of the SAR system. Therefore, in our offloading approach, energy consumption has a higher priority. The computational offloading decision is based on both latency and energy consumption shown in formula (3):

$$L_c = (L_{ps} + L_{Tresult}) - (L_{Tdata} + L_{Pb})$$
$$E_c = E_{ps} + E_{Tresult} - E_{Tdata}$$

$$\begin{cases} \text{offload,} & \text{if } E_c > 0 \text{ and } \dfrac{E_c}{3600 \times V_s \times B_s \times C_s} \geq R_b \times \dfrac{E_{pb}}{3600 \times V_b \times B_b \times C_b} \text{ and } L_c \geq 0. \\ \text{offload,} & \text{if } E_c > 0. \text{ and } \dfrac{E_c}{3600 \times V_s \times B_s \times C_s} \geq R_b \times \dfrac{E_{pb}}{3600 \times V_b \times B_b \times C_b} \text{ and } L_c < 0. \\ \text{local process,} & \text{if } E_c > 0 \text{ and } \dfrac{E_c}{3600 \times V_s \times B_s \times C_s} < R_b \times \dfrac{E_{pb}}{3600 \times V_b \times B_b \times C_b} \text{ and } L_c < 0. \\ \text{local process,} & E_c \leq 0 \end{cases} \tag{3}$$

In case both energy efficiency and latency reduction targeted, the algorithm for computation offloading decision firstly checks the energy consumption as energy has a higher priority. If it is possible to save energy, then computation offloading (big drone) should be done. Else, data should be locally processed (small drone). An exception is a special case where the percentage of energy saved (small drone) is much less than the percentage of energy spending (big drone). It is taken into account that a big drone is responsible for several small drones. Therefore, it is necessary to consider this special case to avoid a bottleneck at a big drone.

5.2. Computation offloading type 2

Similar to the computation offloading type 1, the second computation offloading type from a big drone to an edge server also targets reducing latency or saving energy. In the case that latency reduction is targeted, formula (1) is used. If saving energy is focused, formula (4) is applied.

$$E_c = E_{pb} + E_{Tresultb} - E_{Tdatab} \begin{cases} \text{offload,} & E_c > 0 \\ \text{local process,} & E_c \leq 0 \end{cases} \tag{4}$$

When both conserving energy and latency reduction are considered, one of them should have a higher priority depending on the specification of a big drone and the requirements of the SAR system. For example, if a big drone is just a battery-based drone with a flying time of around 20–25 min, saving energy can have a higher priority. On the other hand, if a big drone uses a large-capacity hydrogen tank as its main power supply, the drone can fly for around 40–50 min. In this case, latency should have a higher priority. The flying time of a small drone is around 20–25 min while the flying time of a big drone can be approximately 30–50 min. Therefore, the bottleneck of the SAR system is small drones, which cannot help to achieve more benefits when saving more energy on a big drone. In our case, since a big drone is a battery-based drone, saving energy has a higher priority than latency. Formula (5) is used for this case.

$$L_c = (L_{pb} + L_{Tresult}) - (L_{Tdata} + L_{Pb})$$
$$E_c = E_{ps} + E_{Tresult} - E_{Tdata}$$
$$\begin{cases} \text{offload,} & E_c > 0 \text{ and } L_c \geq 0 \\ \text{offload,} & E_c > 0 \text{ and } L_c < 0 \\ \text{local process,} & E_c \leq 0 \end{cases} \tag{5}$$

In the proposed IoT system for SAR, videos, images captured with small drones are used as inputs for calculating values presented in formulas (1), (2), and (3). Based on the computational offloading decision-making, images or other information extracted for human detection algorithms will be transmitted from a small drone to a big drone. Similarly, the same process occurs with formulas (1), (4), and (5) in the case of offloading between a big drone and an edge server.

6. System setup

This section presents a setup of different drones (e.g., small and big drones), an edge server, and a blockchain.

6.1. Setup of small drones, big drones, and an edge server

The specifications used in the experiments for drones are shown in Table 4. A cluster of drones in the test setup consists of four small drones and one big drone shown in Fig. 9. In practice, small drones should be equipped with high-resolution RGB and thermal cameras to enable daytime and nighttime operations. The small drones' cost and complexity in the experiments are reduced by only using an RGB camera. The drones have embedded boards and wireless communication modules (e.g., Wi-Fi modules). Due to the limitations of a small drone's light load support, they cannot carry powerful and more heavy embedded boards that often consume considerably more energy. The big drones can be equipped with high-performance embedded boards and support heavier loads that allow higher battery capacity, thus extending flight time. Table 4 shows the estimated prices of components and devices to build these drones, which vary significantly depending on the sensors and cameras placed on the drone. In the implementation, a small drone is inexpensive compared to a big drone since the big drone has more resources, better cameras, and a larger battery capacity.

An edge server used for the experiments has the following specifications AMD Ryzen Threadripper 3960X 24-Core 3.8 GHz processor, two NVIDIA GeForce RTX 3090 GPUs (24 GB VRAM each), 128 GB RAM, 1TB SSD, and Ubuntu 20.04 with Lambda Stack pre-installed with a total cost of about 9.7k Euros. The edge server placed on a SAR boat using mains power could be powered from batteries via a converter.

6.2. Blockchain setup

Since the usage of blockchain for SAR requires different services, a blockchain-based smart contract is a prominent answer for a blockchain-based SAR. In detail, a blockchain-based use case is a transaction-based state machine in which the state machine is according to a series of inputs by which the view of the system transits to states. Therefore, a blockchain-based SAR is based on a transaction-based state machine to form services in SAR. With this consideration, this subsection is about blockchain network facility and setup for two different blockchain platforms, including Ethereum and Hyperledger Fabric.

6.2.1. Ethereum

As the structure, three identical virtual machines represent the Ethereum setup through the aforementioned physical machine Intel(R) Core(TM) i7-8700 CPU @ 3.20 GHz, 3.19 GHz, 32 GB RAM, and 64-bit OS, x64-based processor. Each virtual machine resources consist of 6 processes and 6 GB of memory with an OS x86/64 running the Official Go implementation of the Ethereum protocol, called *geth*, version 1.10.12-stable. The setup of three virtual machines w.r.t three participants in the Ethereum network maintains the system through workflow and consensus. The smart contract deployment is based on Truffle,[2] a tool for a blockchain development environment utilizing EVM.

Ethereum [33] represents a traditional smart-contract blockchain as an order-execute workflow. Based on the consensus concept, the order-execute design is to order transactions before broadcasting and executing those sequentially. Besides, Ethereum, a permissionless blockchain-based smart contract, allows participants to join and leave the system freely. Supporting this type of system, Ethereum utilizes PoW consensus to gain agreements among trustless participants. PoW consensus requires participants to seek a nonce as PoW result in Fig. 1 to prove the correctness of proposals; thus, after finding a valid nonce, the participant

[2] https://trufflesuite.com/

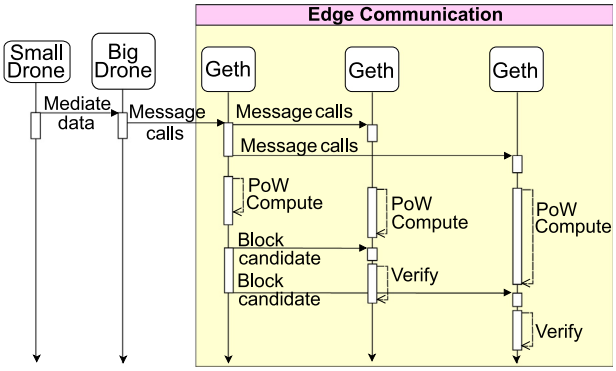

Fig. 4. General sequential diagram for Ethereum workflow.

Table 5
The hardware components for edge servers.

	Information	OS
Docker	Version: 18.09.7	amd64
Orderer	Fabric: 2.2.1, Process: 2, Memory: 2 GB	x86_64
Peer	Fabric: 2.2.1, Process: 4, Memory: 4 GB	x86_64

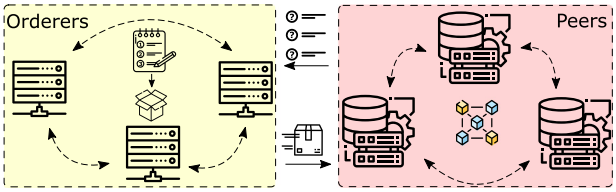

Fig. 5. The design of edge servers' communication with the orderer organization (yellow region) and the peer organization (pink region): (Peers) the set of peers' service maintains the blockchain and smart contract computation, (Orderers) the list of orderers receive endorsed transaction proposals and order those proposals into a block as block candidate. This block candidate is transferred to peers for validation.

generates a block candidate and propagates it to others for verification and approval. As a general view of the Ethereum setup Fig. 4, each big drone handles an externally owned account to call smart contracts as message calls while edge servers maintain a unique blockchain, world state, and smart contracts along with contract accounts. Therefore, each participant running the *geth* application receives message calls from big drones and stores them in the transaction pool. Meanwhile, network participants execute a race condition to gain the permission of block generation. For example, in Fig. 4, the left participant obtains permission due to the first participant finding a suitable nonce. After that, the block candidate wrapped with chronological message calls from the transaction pool is broadcast to other participants for verification and blockchain extension. Note that contract accounts manage smart contracts and interact with other smart contracts through function calls, supporting the decentralized application concept.

6.2.2. Hyperledger Fabric

An array of six virtual machines forms the blockchain network, equally separated into three Orderers and three Peers. The deployment is based on a physical machine that is the same as the machine for the Ethereum setup.

Each edge server has to deploy a service for Orderer or Peer. However, to keep the basic setup that can avoid a single failure from each service, we desire to equally separate three virtual machines for Peer service and the other three virtual machines implementing the Orderer service. In addition, our edge servers' connectivity and data exchange are specified in Fig. 5. Also, each machine's hardware details are shown in Table 5.

From the perspective of the blockchain layer, a permissioned blockchain called Hyperledger Fabric [39] and smart contracts are leveraged. With Fabric, the system obtains a consensus mechanism, smart contract workflow, data model, and cryptographic schemes. As the boat servers maintain blockchain without resource limitation, a full deployment of blockchain is developed. In addition, by leveraging the permissioned blockchain, the system reaches the consensus agreements faster than others and guarantees the users' confidentiality. Fabric proposes execute-order-validate architecture with a key–value data model, certificate authority, and Raft consensus. Following the workflow of Fabric's architecture, the clients, as big drones, at the execution phase, first create and send transaction proposals to specific peers called endorsed peers to execute and gain endorsed transaction proposals. After that, the client forwards the endorsed transaction proposal to orderers to totally order and wrap endorsed transaction proposals into a block candidate at the ordering phase. Then, the orderers broadcast the block candidate to entire peers, which validate at the validation phase before approving it as the novel block. The whole procedure is in Fig. 6, and the edge communication observes in Fig. 5.

6.2.3. Smart contract design

We design smart contracts through our scenario in the interest of smart contracts via decentralized applications. In particular, our design includes three parts, as shown in Fig. 7: a drone object, a rescuer team, a hospital, and a supplier. The drone object represents big drones' information, including the location, the connectivity between small and big drones, and previous data. The rescuer team presents rescuer teams that contain a set of specialists, locations, and previous archives. The hospital contract is for hospitals that receive requirements from rescuer teams and prepare medical equipment for the rescuing process; otherwise, the hospital contract supports an autonomous search for another hospital or suppliers in collaboration to facilitate the mission. For example, when a small drone finds a victim, the information is sent to a big drone. First, the big drone updates its information on the drone object. Then, through the offload computation, the drone object decides to send urgent information to a rescuer team nearby for assistance. At this consideration, if the rescuer team cannot handle and require some other help, the rescuer team continues sending a requirement request to the closest hospital for demands, such as an ambulance, blood suppliers, laboratories, and emergency rooms, or further a request to a potential supplier or another hospital for collaboration. Those smart contracts are implemented for a demonstration setup, and these smart contracts can autonomously send requests to each other through function calls and decision-making in different contexts.

7. Experimental results

This section includes experimental results of computation offloading and blockchain. In the computation offloading subsection, latency and energy consumption results are reported, and the blockchain subsection has the results and comparison of Hyperledger and Ethereum.

7.1. Computation offloading

Embedded boards were used in the experiments, including Jetson Nano, Jetson Xavier NX, and Jetson AGX Xavier. In terms of processing and memory storage, Jetson Nano is the least powerful and consumes less energy, and conversely, Jetson AGX Xavier consumes the most power and has the highest amount of processing capability. Jetson Nano's weight is the lightest while Jetson AGX Xavier's weight is the heaviest. According to our experience and hardware specification, Jetson Nano and Jetson Xavier NX are more suitable for small drones, while Jetson AGX Xavier is for big drones. Comprehensive information about these boards, max power, number of CPUs, max frequency, and operating modes are shown in Table 6. Each Jetson embedded board has several operating modes with different energy consumption and

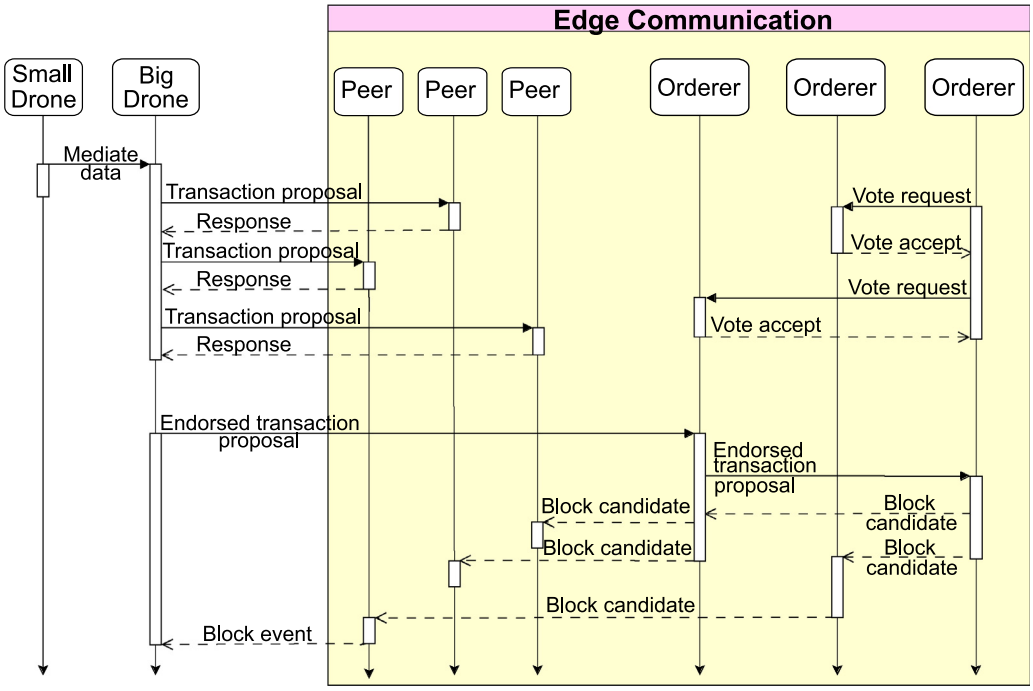

Fig. 6. The sequence diagram of the proposal.

Table 6
The different settings in Jetson Nano, Jetson Xavier NX, and AGX.

Jetson	Xavier	NX							
P	15	15	15	10	10	10	20	20	20
M	0	1	2	3	4	5	6	7	8
CPUs	2	4	6	2	4	4	2	4	6
F	1900	1400	1400	1500	1200	1900	1900	1400	1400

Jetson		AGX	Xavier					Jetson	Nano
P	N/A	10	15	30	30	30	30	15	10	5
M	0	1	2	3	4	5	6	7	0	1
CPUs	8	2	4	8	6	4	2	4	4	2
F	2265.6	1200	1200	1200	1450	1780	2100	2188	1479	918

P = Max Power (W), M = Mode ID, F = Max Frequency (MHz).

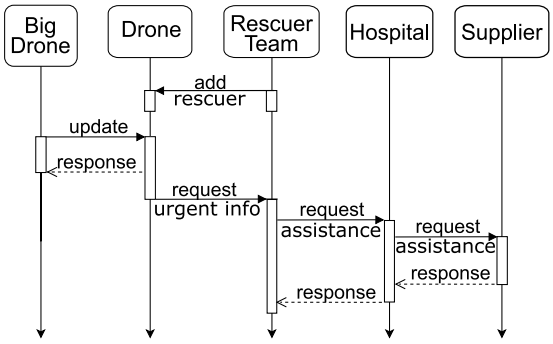

Fig. 7. The sequence diagram of smart contract in the proposal.

computing capabilities. Jetson NX has the fastest inference speed for the models trained with YOLOv4 and YOLOv5 when one of the 20 W operating modes (mode 6, 7, or 8) is used regardless of how many CPU cores are active. Jetson AGX has the best performance in the 30 W operating modes (mode 3, 4, 5, or 6). Use of the Jetson AGX's maximum power mode (mode 0) is not advisable even if there is enough battery capacity since it does not limit power usage. Large current surges occur during object detection using this mode, which could cause a system shutdown. Choosing a power mode that uses at least 4 CPU cores is recommended to handle additional tasks in time, even though there is very little difference in the object detection performance due to the trained models relying more on GPU processing. Therefore, depending on the application, a particular mode should be chosen. Experimental standby power consumption of different boards is as follows: Raspberry Pi 4 B consumes 2.60 W; Jetson Nano consumes 1.10 W; Jetson Xavier NX consumes 4.94 W; Jetson AGX Xavier consumes 5.32 W.

Insights into data transmission between a small drone and a big drone are provided with latency measurements of transferring images from a small drone to a big drone via Wi-Fi. Several packet sizes such as 10k and 65.5 KBytes (Max UDP datagram size) are applied for transmission via the UDP and DTLS protocols. It is noted that transmission distance affects energy consumption and latency results. Particularly, transmission latency and energy consumption increase when distance transmission is further. However, the maximum transmission power is applied for the experiments, and the direct one-to-one transmission speed of Wi-Fi is extremely fast. Therefore, our experiments apply an average transmission distance of 15 m. Correspondingly, the experimental results of energy consumption and latency on the 15-meter transmission distance are still relatively accurate even though the actual distance between drones can be a few hundred meters. The average latency from 100 measurements is reported in Table 7. The results show that sending data with a larger packet size is more desirable, as it can help reduce total transmission latency. In our case, one image captured from a drone's camera is around 5–6 MBytes. Therefore, a larger packet size can be applied in the transmission for most cases. However, when the connection is poor, the smaller packet size can be used.

For protecting data transmitted in internal networks (e.g., from a small drone to a big drone and from a big drone to an edge server), cryptography primitives such as AES-256 and ChaCha20 are used to encrypt the data. The latency results of data encryption and decryption at different embedded boards are shown in Fig. 8. The results show that the encryption and decryption latencies are minimal (i.e., a few

Table 7
Latency for sending packets.

Transfer protocol	Round-trip latency (ms)	Latency (ms)
UDP (10 kB)	7.2	3.6
UDP (65 kB)	10.8	5.4
DTLS 1.2		
AES256-GCM (10 kB)	7.9	4.0
AES256-GCM (65 kB)	22.3	11.2
ChaCha20-Poly1305 (10 kB)	8.3	4.2
ChaCha20-Poly1305 (65 kB)	22.2	11.1
AES256-GCM (new connection + 10 kB	103.9	–
AES256-GCM (new connection + 65 kB	108.2	–

Table 8
Inference speed and energy consumption for different mode settings with YOLOv4 trained model.

	Mode	AVG FPS	Current (A)	Power (W)
Jetson Nano 5 V	0	1.1	1.56	7.80
	1	0.8	1.15	5.75
Jetson Xavier NX 19 V	0	5.7	0.72	13.68
	1	5.8	0.76	14.44
	2	5.8	1.02	19.38
	3	5.4	0.81	15.39
	4	5.4	0.80	15.20
	5	3.9	0.78	14.82
	6	6.8	1.25	23.75
	7	6.9	1.24	23.56
	8	6.9	1.26	23.94
Jetson AGX Xavier 19 V	0	10.9	2.05	38.95
	1	2.7	0.58	11.02
	2	6.2	0.87	16.53
	3	8.1	1.12	21.28
	4	8.2	1.12	21.28
	5	8.1	1.18	22.42
	6	8.0	1.14	21.66
	7	6.2	0.93	17.57

milliseconds) for 16-byte block size. Particularly, using AES256-GCM cipher to encrypt and decrypt 300 MB of data with Jetson NX takes 2.97 ms, and with ChaCha20-Poly1305 cipher, it would take 4.96 ms. Compared with a total latency, the latency caused by applying cryptography algorithms is much smaller. Therefore, it is recommended to apply the cryptography algorithms (e.g., AES256 and ChaCha20) for the SAR system, especially for the internal networks (e.g., between small drones and big drones, between big drones and edge servers). Secret/primary keys for encrypting and decrypting can be distributed between the participants (e.g., small drones and big drones) in advance before each mission. These pre-shared keys can be a list of keys in which each key can be used once in a specific period, such as every 10 min.

In our experiments, two widely used object detection models, including YOLOv4 [14] and YOLOv5 [15], were used to detect humans from images. The YOLOv4 model consists of the backbone (CSPDarknet53 [40]), neck (SPP [41] and PAN [42]), and head (YOLOv3 [43]). The pre-trained model based on YOLOv4 was applied for the SAR human detection. The YOLOv5 model was introduced by Glenn Jocher using Pytorch framework [15]. The videos related to SAR missions are captured by drones flying 60 m above the ground and used as the input for the models. The videos have 60 frames per second and a 1920 $\times$ 1080 resolution. In the videos, a volunteer performs different actions, including imitations of drowning. Before feeding models, each video frame is cropped and resized to tiles of 640 $\times$ 544 resolution. The trained models are then deployed at different embedded boards.

Table 9 shows the performance of different boards when using the YOLOv5 algorithm. Raspberry Pi 4 and Intel Up Squared power consumption is around 6 W and 7 W. These both utilize CPUs for inferring objects which causes them to use many resources. When this object detection model is used, they also have more significant processing

Table 9
Inference speed and energy consumption for different mode settings with YOLOv5 trained model.

	Mode	AVG FPS	Current (A)	Power (W)
Raspberry Pi 4 B 5 V	N/A	0.23	1.21	6.05
Intel Up Squared 5 V	N/A	0.37	1.48	7.40
Jetson Nano 5 V	0	2.8	1.53	7.65
	1	1.8	1.05	5.25
Jetson Xavier NX 19 V	0	12.1	1.02	19.38
	1	12.2	1.06	20.14
	2	12.2	1.08	20.52
	3	9.8	0.62	11.78
	4	9.8	0.60	11.40
	5	6.8	0.56	10.64
	6	12.8	1.15	21.85
	7	12.8	1.19	22.61
	8	12.8	1.22	23.18
Jetson AGX Xavier 19 V	0	20.8	1.27	24.13
	1	4.7	0.55	10.45
	2	10.9	0.70	13.30
	3	14.4	0.78	14.82
	4	13.9	0.94	17.86
	5	14.4	0.98	18.62
	6	14.4	1.04	19.76
	7	10.9	0.77	20.90

latency. Nvidia's Jetson Nano is also suitable to use with a small drone, and it achieves almost three frames per second with YOLOv5 trained model when using the higher power mode, and it consumes 7.65 W when the object detection model is being used. Jetson Nano's 5 W power mode is useful when the device is mostly idle due to significantly lower performance. The average inference speed is slower in the model trained with the YOLOv4 algorithm, shown in Table 8.

In the experiments, computation offloading focuses on type 1 from a small drone to a big drone and computation offloading type 2 from a big drone to an edge server. In computation offloading type 1, both YOLOv4 and YOLOv5 are applied with three policies: latency-based decision-making, energy-based decision-making, and latency-and-energy-based decision-making. Based on the policy, the appropriate formulas (i.e., formula (1)–(3)) are applied. The computation offloading results when using YOLOv4 and YOLOv5 are shown in Table 10 and Table 11, respectively. It is noted that the latency results rely on transmission latency which depends on many factors such as Wi-Fi technology, transmission environment, transmission distance, and noise. Therefore, there will be the bias in the measurements. Fortunately, these factors less affect our system much since drones mainly search in the open-space sea.

In the experiments, the average Wi-Fi transmission speed is around 140 Mbps with 2.4 GHz 802.11n Wi-Fi. Accordingly, the transmission latency is around 280–350 ms for transmitting an image size of 5–6 MBytes. The measurement latency results are sensitive to the received signal level and link quality between the devices.

As discussed, the energy-based policy for computation offloading relies on also battery capacity and battery level. Therefore, the computation offloading decision may change when the battery level changes. The experiment assumes that the battery level is 50% for both small drones and big drones. Results shown in Table 10 and Table 11 are impacted by these parameters. The results show that offloading helps reduce latency and saves energy in most experimental cases. When specifications of small drones and big drones are similar, it may be more beneficial in terms of latency to process data locally at a small drone.

The results of computation offloading type 2 from a big drone to an edge server are shown in Table 12. The experiments are carried out with two models, including Yolov4 and Yolov5. The results show that it is more beneficial in terms of latency to process images locally at a big drone. The main reason is that an embedded board of a big drone is powerful enough to process images, while the transmission latency is quite significant for sending an image (e.g., 5–6 MBytes). In the case of

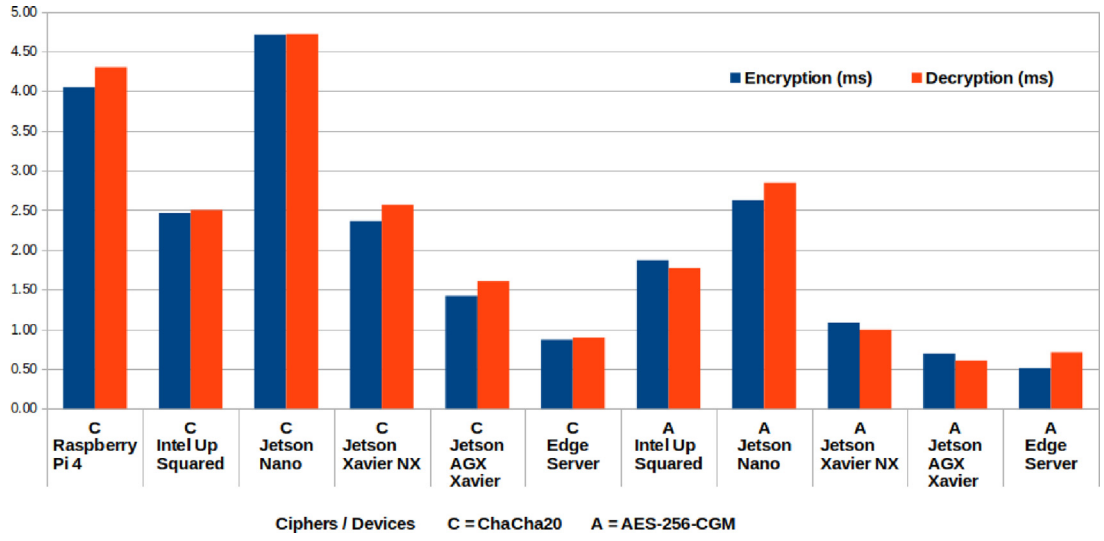

Fig. 8. Decryption and Encryption at different boards.

(a) Small drone (b) Big drone (Customized S1000) (c) Cluster of drones (d) An image captured by a drone (Autel robotics Evo II) with a 6K resolution camera at 60m height

Fig. 9. Examples of small and big drones, and a cluster of drones.

Table 10
Decision-making of computation offloading type 1 from a small drone to a big drone when using the model based on YOLOv4.

SD board	BD board	Latency-based policy	Energy-based policy battery level = 50%	Latency&Energy-based policy
Jetson Nano	Xavier NX	Offload save 500–580 ms	Offload save 5.5-6.2.J at SD E_{total} = 3.4-3.7 J	Offload
Jetson Nano	AGX Xavier	Offload save 570–650 ms	Offload save 5.5-6.2 J at SD E_{total} = 3.6 J	Offload
Xavier NX	AGX Xavier	Local processing save 200–240 ms	Offload save 1.2-1.4 J at SD spent 2.62 J at BD	Local processing(if latency) Offload (if energy)

BD = Big Drone, SD = Small Drone.
Total energy saving $E_{total} = E_{ps} + E_{tResult} - E_{tis} - E_{pb}$.
Symbol descriptions are in Table 3.

energy saving, it is more beneficial to offload a task from a big drone to an edge server.

7.2. Blockchain performance

The throughput is calculated with different situations for Ethereum and Hyperledger Fabric as the blockchain performance. In detail, the results for the two platforms are displayed in Table 13 and Table 14. Notably, the columns show different types of requests, including update information (update info), query information (query info), and request calling a chain of connected smart contracts (call contracts); meanwhile, the rows mention the diversity of capacity.

7.2.1. Ethereum result

The result for Ethereum performance is illustrated in Table 13 based on three main kinds of requests submitted to the system. As a result of throughput, a common observation is about the increase of capacity at requests corresponding to the decline of the number of requests the system can handle per second. For example, with 16 kB, 32 kB, and 65 kB, the number of requests the system can handle per second with Update Info reducing from 22.467 to 6.158, Query Info declining from 314.685 to 85.146, and Call Contract dropping from 18.426 to 5.172. In comparison among request types, Tx/s with Call Contracts is lower than Update Info in the same capacity due to data update execution after three smart contract calls. Also, Query Info gains much high Tx/s than

Table 11
Decision-making of computation offloading type 1 from a small drone to a big drone when using the model based on YOLOv5.

SD board	BD board	Latency-based policy	Energy-based policy battery level = 50%	Latency&Energy-based policy
Jetson Nano	Xavier NX	Offload saves 70–150 ms	Offload saves 2.3-2.5 J at SD E_{total} = 0.7-0.9 J	Offload
Jetson Nano	AGX Xavier	Offload saves 120–200 ms	Offload saves 2.3-2.5 J at SD E_{total} = 1.1-1.3 J	Offload
Xavier NX	AGX Xavier	Local saves 200–250 ms	Offload saves 0.1-0.5 J at SD spent 1.2 J at BD	Local if latency Offload if energy

Symbol descriptions are in Table 3.

Table 12
Decision-making of computation offloading type 2 from a big drone to an edge server when using the model based on YOLOv4 and YOLOv5, (L)=Local (O)=Offload.

Model	BD	Latency	Energy	L&E
YOLOv4	Xavier NX	Local saves 130 ms	Offload saves 2.78 J at BD	(L) if latency or (O) if energy
YOLOv4	AGX Xavier	Local saves 180 ms	Offload saves 2.57 J at BD	(L) if latency or (O) if energy
YOLOv5	Xavier NX	Local saves 210 ms	Offload saves 1.2 J at BD	(L) if latency or (O) if energy
YOLOv5	AGX Xavier	Local saves 230 ms	Offload saves 0.9 J at BD	(L) if latency or (O) if energy

Symbol descriptions are in Table 3.

Table 13
The number of transaction per second (Tx/s) that the system can handle in different capacities and request types in Ethereum platform.

Capacity	Update info	Query info	Call contracts
16 kB	22.467	314.685	18.4267
32 kB	8.023	173.745	10.899
65 kB	6.158	85.146	5.172

Table 14
The number of transaction per second (Tx/s) that the system can handle in different capacities and request types in Hyperledger platform.

Capacity	Update info	Query info	Call contracts
16 kB	2.969	2.906	2.704
32 kB	2.343	2.809	2.257
65 kB	1.526	1.990	1.519

the others since the request is to retrieve data from the local virtual machine.

7.2.2. Hyperledger result

The Hyperledger Fabric blockchain setup experiment is about the throughput with the same idea for request types like Ethereum's experiment, such as Update Info, Query Info, and Call Contracts. Particularly, after deploying smart contracts, the experiment evaluates how many transactions or requests the system can process in a second. The results for this consideration displays in Table 14. Observation from Table 14 shows the big drop in the number of transactions per second the system can handle from 32 kB to 65 kB as 2.343 to 1.526, 2.809 to 1.990, and 2.257 to 1.519 related to Update Info, Query Info, and Call Contracts, respectively. Another interesting point is the difference between the three request types. In detail, the update info's results gain better performance than call contracts requests since the Call Contracts requests need to call a list of three contracts before updating information. Also, the comparison between Update Info and Query Info is interesting, with the result difference at 16 kB capacity from the others. Actually, at 16 kB capacity, the Update Info achieves a better performance than Query Info's result; nevertheless, in the rest of other capacities, Update Info's results obtain lower efficiency than Query Info. Therefore, it can be said that with the low capacity, the throughput of update requests is higher than query information, but the opposite conclusion is swapped if the capacity increases. The difference between Update Info and Query Info results is from communication and data extraction mechanisms. In particular, updating information in the blockchain system requires validation, communication, and agreement among participants; meanwhile, query information is to extract information handled by Peers instead of communication and validation tasks.

7.2.3. Comparison between Ethereum and Hyperledger

In comparing two blockchain platforms, the experiments mention the vast differences between these considerations with a permissioned blockchain — Hyperledger Fabric, and a permissionless blockchain — Ethereum. Remarkably, at Update Info, Ethereum performance achieves a much higher Tx/s (22.467 with 16 kB request) than Fabric results at each different request capacity with 2.969 Tx/s at the same type request. However, with the variety of capacity requests, Fabric results keep steady, while once the capacity in requests, Ethereum performance declines dramatically. For example, with Update Info at 16 kB, 32 kB, and 65 kB, Fabric results are 2.696, 2.343, and 1.526 Tx/s, whereas Ethereum performance is 22.467, 8.023, and 6.158 Tx/s. This observation is similar to Call Contract requests and Query Info. Besides, the massive gap between Hyperledger Fabric and Ethereum is viewed in Query Info with 2.906 and 314.685 Tx/s, although the query is to retrieve information.

A case study is analyzed to provide a clear comparison among those platforms. In the beginning, the participant identity is required in Hyperledger Fabric, but this is not mandatory with Ethereum; hence, the system data is observable in Ethereum, but not in the case of Hyperledger Fabric. With the identity of each participant, the Hyperledger Fabric controls data to access better than Ethereum. For example, during the mission of SAR to find a victim, a big drone captures the information and then sends it to the closest edge server as a blockchain participant with the drone object. At this phase, the Hyperledger Fabric participant running *peer service* receives requests and executes them before replying to the big drone endorsed transaction proposals that are then sent to *orderer service* by the big drone. At Ethereum, the participant receives the request and verifies the correctness before storing it in the transaction pool and propagating it to other participants. Parallel, Hyperledger Fabric participants seek a leader with *orderer services*, similar to Ethereum participants. Once finding the leader for the current consensus round, at both platforms, the leader wraps transactions from their transaction

pool to be a block candidate before broadcasting it to other participants for validation and attaching to the blockchain. Once participants decide to add the block candidate to the current blockchain, transactions or message calls are operated sequentially. Hence, the rescuer team smart contract receives the request and retrieves a possible team to help the victim. Also, the rescuer team considers the team capable of handling the victim, or the contract should generate a message call to the hospital contract with possible requirements and submit it to the blockchain. This message call is similar to the transaction above of finding a victim. The new transaction is put into a transaction pool and waited for confirmation from blockchain participants before executing. Once the message is executed, the hospital contract operates to find a suitable hospital that can provide for the need of the rescue team.

In this case, using hyperledger is more efficient in terms of authentication, authorization, and performance since the participants of the private network require identities. Thus, the system based on a private blockchain guarantees identification and access control for the system, especially in the SAR context.

Another consideration is the connection between system requirements and the performance of blockchain. A possible answer is unclear since the matching or the link requires the detail of the amount of data transferred from drones. For example, drones send the frequency of images or videos to the system and offload tasks. Also, the situation depends on the period of a year. For example, in the summer, people intend to go to the sea or beach, which requires high throughput to ensure the qualify of SAR. However, the detail for the throughput depends on the consensus and the capacity of data sent to the blockchain. In this consideration, the data is mainly images supporting image analyzes for detecting victims and recognizing the rescue process.

7.3. Security satisfaction

The proposed system satisfies all the security requirements (from R1 to R10), as shown in Table 15. Regarding R1-confidentiality and R2-integrity, data is protected and encrypted with AES-128 and ChaCha20Poly1305 before being transmitted over a network (e.g., from a small drone to a big drone, and from a big drone to an edge server). For ensuring R3-authentication and R4-authorization and access control, strong passwords, WPA3, a firewall together with a whitelist storing MAC addresses of authorized drones are applied. The list of strong passwords is applied in which each password is used once every period (e.g., 10–15 min). The passwords can be updated before every SAR mission. The system uses different methods such as passwords and text messages to verify a party/person who would like to access the system. Data at edge servers and cloud servers are categorized and classified. Only authorized persons with suitable levels can access specific data corresponding to their levels. In the case of R5-privacy, GDPR and necessary information related to the victim data are transparently discussed between a rescue team member and a victim. The right to use the acquired data will be collected. Depending on the case, the data can only be used for SAR missions, healthcare entities (e.g., hospitals), or researchers. If a victim is unconscious, the closest/registered victim relative is asked for the right to use the data.

In addition, the utilization of the aforementioned security approaches together with Blockchain technology enables R6-trust in collaboration among parties via immutability and R7-transparency of system state. In detail, as a traditional platform builds the connectivity among the system's entities, blockchain technology replaces the man in the middle to keep information becoming transparent and immutable, enabling trust in communication.

For R8-availability, the information is distributed to any control station via the distribution of blockchain information among participants; hence, the proposed system enables the availability of any possible requests from authorized participants. Regarding R9, the smart contract concept enables automaticity via the connectivity among pre-definitions of different contracts as services. The contract is automatically activated when the requirements are fulfilled to perform desired tasks. The R10-tolerance is fulfilled with the ability to replace crashed components; for example, if a rescuer team is on a mission, another potential rescuer team can be assigned to address the rescue mission; meanwhile, in the case of a crashed component (e.g., a malfunctioned big drone), the system autonomously points the replacement.

Table 15
Reflection of the proposal to security requirements of a SAR system.

R1	R2	R3	R4	R5	R6	R7	R8	R9	R10
X	X	X	X	X	X	X	X	X	X

Abbreviations follows Section 3.2.

State-of-the-art security approaches for drone-based/IoD systems [25–31] have many security limitations shown in Table 2. Most of them only satisfy from R1 to R4 requirements, while other requirements (from R5 to R10) are not meticulously considered. In contrast, the proposed system outperforms to satisfy all ten requirements (from R1 to R10) shown in Table 15.

8. Conclusion and future work

This paper presented an advanced IoD system utilizing edge computing and blockchain technology to overcome the limitations of the existing drone-based systems for SAR. The proposed system consists of a swarm of drones and edge servers providing advanced edge services and edge-AI to search and detect humans automatically from images captured by drones. In addition, computation offloading has been comprehensively investigated and presented. Also, the proposed system encompasses blockchain technology and security approaches to fulfill security requirements such as integrity, authentication, trust, transparency, traceability, authorization, access control, and tolerance. Two blockchain types were applied and compared, including Ethereum and Hyperledger, respectively, public and private. Smart contracts were also designed and implemented to facilitate automaticity. The results showed that the presented system could offer advanced edge services for improving the energy efficiency of drones and the QoS of SAR missions. Besides, using Hyperledger Fabric to form the proposal expects to satisfy SAR's security requirements. Interference in communication bandwidth at the edge can be partially averted using another big drone as a repeater. A rescue team can search many large areas simultaneously with minimal time by applying the proposed system. It helps increase the possibility of finding many missing people or victims simultaneously. Hence, the proposed system's architecture could be a promising approach for SAR, primarily for maritime SAR missions.

Although the complete system, from swarms of drones and edge servers to the blockchain network and applications, had been built, the experiments were still conducted in the lab and controlled environments. Therefore, many aspects of drone and transmission reliability, such as noise, interference, wind speed, and harsh conditions, had not been adequately considered. In addition, although images were captured by small drones flying 60 m above the ground, the environmental conditions were still appropriate e.g., without strong wind. Accordingly, future works should consider separate contexts to improve measurement accuracy. Additionally, although the results show that computation offloading is applied, and the operating time of small drones can be prolonged, the extended flying time is not much, as most drone energy is used for running motors. Hence, to achieve a high level of energy efficiency, it is recommended that computation offloading should be used together with techniques or approaches that help reduce the energy consumption of electric motors. Different distances from drones and flying speed will be considered together with the energy consumption model for offloading in future work. As another consideration regarding decentralization, blockchain reduces the system's performance. Thus, the subsequent work is to optimize blockchain usage to enhance performance. As the blockchain is a technology to form trust

among participants, the performance of blockchain-based systems is a barrier to enhancing and replacing centralized systems with the usage of blockchain.

CRediT authorship contribution statement

Tri Nguyen: Methodology, Writing – original draft, Editing, Visualization, Software, Formal analysis, Validation. **Risto Katila:** Methodology, Writing – original draft, Editing, Visualization, Software, Formal analysis, Validation. **Tuan Nguyen Gia:** Conceptualization, Methodology, Writing – original draft, Visualization, Investigation, Reviewing and editing, Formal analysis, Supervision.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Tri Nguyen reports administrative support and article publishing charges were provided by University of Oulu. Tri Nguyen reports financial support and administrative support were provided by University of Oulu Infotech Oulu. Tri Nguyen reports financial support and administrative support were provided by Academy of Finland. Tri Nguyen reports financial support was provided by Nokia Foundation. Tri Nguyen reports financial support was provided by Tauno Tönning Foundation.

Data availability

The data that has been used is confidential.

Acknowledgments

Tri Nguyen is supported by TrustedMaaS project by the Infotech institute of the University of Oulu, Finland, Nokia Foundation, Finland, Tauno Tönning Foundation, Finland, and Academy of Finland, 6G Flagship program (grant 346208).

References

[1] D.P. Coppola, Introduction to International Disaster Management, Elsevier, 2006.
[2] J.P. Queralta, et al., Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision, IEEE Access 8 (2020) 191617–191643.
[3] J.P. Queralta, et al., Autosos: Towards multi-uav systems supporting maritime search and rescue with lightweight AI and edge computing, 2020, arXiv preprint arXiv:2005.03409.
[4] T. Nguyen, et al., A novel internet-of-drones and blockchain-based system architecture for search and rescue, in: 2021 IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems, MASS, IEEE, 2021, pp. 278–288.
[5] R. Katila, et al., Analysis of mobility support approaches for edge-based IoT systems using high data rate bluetooth low energy 5, Comput. Netw. 209 (2022) 108925.
[6] T.N. Gia, et al., Exploiting LoRa, edge, and fog computing for traffic monitoring in smart cities, in: LPWAN Technologies for IoT and M2M Applications, Elsevier, 2020, pp. 347–371.
[7] T.N. Gia, et al., Artificial intelligence at the edge in the blockchain of things, in: International Conference on Wireless Mobile Communication and Healthcare, Springer, 2019, pp. 267–280.
[8] A. Alalawi, et al., Cloud computing resources: Survey of advantage, disadvantages and pricing, in: 2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy, ICDABI, IEEE, 2020, pp. 1–6.
[9] O. Cheikhrouhou, et al., Blockchain for the cybersecurity of smart city applications, 2022, arXiv preprint arXiv:2206.02760.
[10] A. Nawaz, et al., Edge AI and blockchain for privacy-critical and data-sensitive applications, in: 2019 Twelfth International Conference on Mobile Computing and Ubiquitous Network, ICMU, IEEE, 2019, pp. 1–2.
[11] V.K. Sarker, et al., Lightweight security algorithms for resource-constrained IoT-based sensor nodes, in: ICC 2020-2020 IEEE International Conference on Communications, ICC, IEEE, 2020, pp. 1–7.
[12] H. Nguyen, T. Nguyen, T. Leppänen, J. Partala, S. Pirttikangas, Situation awareness for autonomous vehicles using blockchain-based service co-operation, in: X. Franch, G. Poels, F. Gailly, M. Snoeck (Eds.), Advanced Information Systems Engineering, Springer International Publishing, Cham, 2022, pp. 501–516.
[13] T.H. Nguyen, J. Partala, S. Pirttikangas, Blockchain-based mobility-as-a-service, in: 2019 28th International Conference on Computer Communication and Networks, ICCCN, 2019, pp. 1–6, http://dx.doi.org/10.1109/ICCCN.2019.8847027.
[14] A. Bochkovskiy, et al., Yolov4: Optimal speed and accuracy of object detection, 2020, arXiv preprint arXiv:2004.10934.
[15] G. Jocher, K. Nishimura, T. Mineeva, R. Vilariño, Yolov5, 2020, Code Repository https://github.com/ultralytics/yolov5.
[16] X. Hou, et al., Fog based computation offloading for swarm of drones, in: ICC 2019-2019 IEEE International Conference on Communications, ICC, IEEE, 2019, pp. 1–7.
[17] K. Jo, J. An, J. Jung, H. Min, An offloading decision scheme for a multi-drone system, in: 17th IIE International Conference on Computer, Electrical, Electronics and Communication Engineering, 2017, pp. 1–6.
[18] C. Qu, et al., Dycoco: A dynamic computation offloading and control framework for drone video analytics, in: 2019 IEEE 27th International Conference on Network Protocols, ICNP, IEEE, 2019, pp. 1–2.
[19] D. Chemodanov, et al., Policy-based function-centric computation offloading for real-time drone video analytics, in: 2019 IEEE International Symposium on Local and Metropolitan Area Networks, LANMAN, IEEE, 2019, pp. 1–6.
[20] R. Valentino, W.-S. Jung, Y.-B. Ko, Opportunistic computational offloading system for clusters of drones, in: 2018 20th International Conference on Advanced Communication Technology, ICACT, IEEE, 2018, pp. 303–306.
[21] D. Callegaro, M. Levorato, Optimal computation offloading in edge-assisted uav systems, in: 2018 IEEE Global Communications Conference, GLOBECOM, IEEE, 2018, pp. 1–6.
[22] W. Lin, et al., Energy-efficient computation offloading for UAV-assisted MEC: A two-stage optimization scheme, ACM Transactions on Internet Technology (TOIT) 22 (1) (2021) 1–23.
[23] L. Li, X. Wen, Z. Lu, W. Jing, An energy efficient design of computation offloading enabled by UAV, Sensors 20 (12) (2020) 3363.
[24] A. Koubâa, et al., Deepbrain: Experimental evaluation of cloud-based computation offloading and edge computing in the internet-of-drones for deep learning applications, Sensors 20 (18) (2020) 5240.
[25] Z. Lv, et al., Analysis of using blockchain to protect the privacy of drone big data, IEEE Netw. 35 (1) (2021) 44–49.
[26] Y. Wu, et al., Blockchain-based privacy preservation for 5G-enabled drone communications, IEEE Netw. 35 (1) (2021) 50–56.
[27] A. Islam, et al., A blockchain-based artificial intelligence-empowered contagious pandemic situation supervision scheme using internet of drone things, IEEE Wirel. Commun. 28 (4) (2021) 166–173.
[28] A. Gumaei, et al., Deep learning and blockchain with edge computing for 5G-enabled drone identification and flight mode detection, IEEE Netw. 35 (1) (2021) 94–100.
[29] Z. Chang, et al., Blockchain-empowered drone networks: Architecture, features, and future, IEEE Netw. 35 (1) (2021) 86–93.
[30] A. Yazdinejad, et al., Enabling drones in the internet of things with decentralized blockchain-based security, IEEE Internet Things J. 8 (8) (2021) 6406–6415.
[31] S. Luo, et al., Blockchain-based task offloading in drone-aided mobile edge computing, IEEE Netw. 35 (1) (2021) 124–129.
[32] R.C. Merkle, A digital signature based on a conventional encryption function, in: Conference on the Theory and Application of Cryptographic Techniques, Springer, 1987, pp. 369–378.
[33] G. Wood, et al., Ethereum: A secure decentralised generalised transaction ledger, in: Ethereum Project Yellow Paper, 2014, pp. 1–32.
[34] V. Buterin, A next-generation smart contract and decentralized application platform, White Pap. 3 (37) (2014).
[35] T.N. Gia, et al., Edge AI in smart farming IoT: CNNs at the edge and fog computing with LoRa, in: 2019 IEEE AFRICON, IEEE, 2019, pp. 1–6.
[36] A.M. Rahmani, et al., Exploiting smart e-health gateways at the edge of healthcare internet-of-things: a fog computing approach, Future Gener. Comput. Syst. (2017).
[37] D. Ongaro, J. Ousterhout, In search of an understandable consensus algorithm, in: 2014 {USENIX} Annual Technical Conference ({USENIX} {ATC} 14), 2014, pp. 305–319.
[38] F. Tschorsch, B. Scheuermann, Bitcoin and beyond: A technical survey on decentralized digital currencies, IEEE Commun. Surv. Tutor. 18 (3) (2016) 2084–2123.
[39] E. Androulaki, et al., Hyperledger fabric: a distributed operating system for permissioned blockchains, in: Proceedings of the Thirteenth EuroSys Conference, 2018, pp. 1–15.
[40] C.-Y. Wang, et al., CSPNet: A new backbone that can enhance learning capability of CNN, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 390–391.

[41] K. He, et al., Spatial pyramid pooling in deep convolutional networks for visual recognition, IEEE Trans. Pattern Anal. Mach. Intell. 37 (9) (2015) 1904–1916.
[42] W. Liu, et al., Ssd: Single shot multibox detector, in: European Conference on Computer Vision, Springer, 2016, pp. 21–37.
[43] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, 2018, arXiv preprint arXiv:1804.02767.

Tri Nguyen was born in Ho Chi Minh, Vietnam, in 1993. He received the B.Sc. degree in computer science from the University of Information Technology — Vietnam National University, Vietnam in 2015, and the M.Sc. degree in computer science from the University of Pisa, Italy, in 2018. Since 2018, he has been a doctoral student in the Center for Ubiquitous Computing, University of Oulu. His research interests include distributed systems, blockchain technology, and information security.

Risto Katila received a Bachelor of Engineering degree in electronics specializing in communication systems from Turku University of Applied Sciences in 2015 and a master's degree in information and communication technology in 2021 from the University of Turku. His research interest includes wireless networks and the Internet of Things.

Dr. Tuan Nguyen Gia received the Ph.D. degree in technology. He has published more than 60 international peer-reviewed articles. He is working on edge and fog computing, and edge-AI. His research interest is e-health, drones, autonomous and embedded systems, smart cities, and blockchain. He is also working on reconfigurable computing with FPGA and CGRA, energy efficiency of wearable devices and remote monitoring systems. He is also a Topic Editor of Vehicles, Future Internet and Sensors journals.