Blockchain-Based Trusted Traffic Offloading in Space-Air-Ground Integrated Networks (SAGIN): A Federated Reinforcement Learning Approach

Fengxiao Tang, *Member, IEEE*, Cong Wen, *Student Member, IEEE*, Linfeng Luo, *Student Member, IEEE*, Ming Zhao, *Member, IEEE*, and Nei Kato, *Fellow, IEEE*

***Abstract*— In the future era of intelligent networks, communication technology and network architecture need to be further developed to provide users with high-quality services. The Space-Air-Ground Integrated Networks (SAGIN) is seen as a potential architecture to provide ubiquitous communication and drive the era of the intelligent global network. The space and air segments in SAGIN can assist in offloading traffic from the ground segment. However, in a highly dynamic and heterogeneous network like SAGIN, offloading decisions are easily affected by the incorporated/malicious nodes. How to ensure security and improve network performance becomes a critical problem. In this paper, we address the above problem by jointly using blockchain and federated reinforcement learning (FRL). Firstly, we propose a blockchain-based secure federated learning framework that combines topology information chain and model chain to assist traffic offloading. Then, we propose a node security evaluation and an enhanced practical byzantine fault tolerance (EPBFT) algorithm to secure the traffic offloading process. Furthermore, we describe the traffic offloading problem as a Markov decision problem (MDP) and employ the Blockchain-based Federated Asynchronous Advantage Actor-Critic (BFA3C) algorithm to solve this problem. Finally, the simulation results show that the BFA3C-based algorithm used in SAGIN with/without malicious nodes achieves superior performance in terms of latency and security.**

***Index Terms*— Space-air-ground integrated networks (SAGIN), blockchain, malicious node, traffic offloading, federated learning, reinforcement learning.**

I. Introduction

In RECENT years, with the implementation of 5G, terrestrial wireless communication technology has developed rapidly. Aims to future intelligent and ubiquitous network, 6G is proposed to support five future application scenarios: Enhanced Mobile Broadband Plus (eMBB-Plus), Big Communications (BigCom), Secure Ultra-Reliable Low-Latency Communications (SURLLC), Three-Dimensional Integrated Communications (3D-InteCom), and Unconventional Data Communications (UCDC) [1]. However, it is challenging to meet the ultra-high Quality of Service (QoS) requirement of those applications with terrestrial wireless technology alone. It is expected that by 2030 over 500 billion IoT devices will be in use with the functions of sensing, computing, and communications, in which a surging amount of data and information exchanges are carried out among different IoT devices [2]. However, due to limited coverage and capacity, the terrestrial network segment may not be able to provide stable network access to users in remote/non-terrestrial areas, let alone meet the above service requirements. In view of the above problems, utilizing modern information network technologies and interconnecting space, air, and ground network segments, the space-air-ground integrated network (SAGIN) has attracted much attention from academia to industry [3], [4].

Due to its advantages of integrating three network segments, SAGIN has significantly improved its network coverage, capacity, and flexibility. Thanks to its superior performance, SAGIN plays a vital role in the fields of military [5], emergency disaster relief [6], and intelligent transportation [7]. While SAGIN offloads traffic/computation load and dramatically enhances the performance of the terrestrial network segment, it also brings some challenges. The use of satellites and UAVs to assist communications brings high mobility and heterogeneity issues, which are hard to accurately describe with static models and fixed rules. The traditional traffic offloading method based on fixed rules cannot play a good role in SAGIN. Traditional traffic offloading schemes are aimed at static, homogeneous networks, and they rarely consider the real-time link state and changing topology of the network [8]. In addition, cross-segment communication and information interaction will bring some security problems in a highly heterogeneous network such as SAGIN [9].

Due to the huge network space of SAGIN, large signaling overhead and user privacy, it is challenging to collect global network information in real-time. However, the traditional centralized machine learning techniques require global data, which is becoming a bottleneck of large-scale implementation due to the transmission overhead and significantly increased privacy concerns of raw training data exchange [10]. As a distributed machine learning technology, federated learning

Manuscript received 16 March 2022; revised 28 June 2022; accepted 30 June 2022. Date of publication 13 October 2022; date of current version 22 November 2022. This work was supported in part by the Intelligent annotation and fine-grained recognition of large-scale multimodal medical behavior belong to 2030 Innovation Megaprojects (to be fully launched by 2020) through New Generation Artificial Intelligence Project under Grant 2020AAA0109602, in part by the Key Research and Development Program of Xinjiang Autonomous Region under Grant 2021B01002, and in part by the National Key Research and Development Program of China under Grant 2021ZD0140301. *(Corresponding author: Fengxiao Tang.)*

Fengxiao Tang, Cong Wen, Linfeng Luo, and Ming Zhao are with the School of Computer Science and Engineering, Central South University, Changsha 410083, China (e-mail: tangfengxiao@csu.edu.cn; congwen@csu.edu.cn; luolinfeng@csu.edu.cn; meanzhao@csu.edu.cn).

Nei Kato is with the Graduate School of Information Sciences (GSIS), Tohoku University, Sendai 980-8579, Japan (e-mail: kato@it.is.tohoku.ac.jp).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSAC.2022.3213317.

Digital Object Identifier 10.1109/JSAC.2022.3213317

solves the problem of information silos. Recently, researchers have discussed the application of federated learning in wireless networks [11], [12], [13]. In an extensive network such as SAGIN, federated learning technology does not require knowledge of global network information, which protects user privacy and reduces the communication overhead of transmitting raw data. However, some reecent researches show that an attack on the central server would destroy the aggregation of the model and gain access to the user's privacy during the update process. Although some research works focus on local model security in federated learning [14], [15], they all assume the central server is secure and does not consider the central server intrusion with falsified data. Therefore, it is a huge challenge to address the problem of attacks on federated learning during the upload and distribution of models and the secure sharing of network topology information to cope with traffic offloading in such large network structures as SAGIN.

Blockchain has received much attention from researchers due to its transparency, decentralization, and security features in the past few years. The security problem of decentralization of federated learning can be effectively solved by storing and sharing models on the blockchain. Zhou et al. [16] proposed a blockchain-based distributed-learning framework which utilizes blockchain to protect global model. Furthermore, Li et al. [17] proposed a blockchain-based federated learning framework with committee consensus to avoid central server risks and optimize network transmission. Blockchain and distributed ledger technology is one of the most disruptive technology enablers to address most of the current limitations and facilitate the functional standards of 6G [18]. Intuitively, using blockchain technology to establish trusted interaction between untrusted nodes in the network enables any two devices in SAGIN to safely complete data interaction without relying on a trusted third party should be a potential way to solve the central server intrusion as mentioned above.

However, the above works assume the network is static, and none of them consider the dynamic transmission overhead issue, which is unsuitable for the dynamic and heterogeneous SAGIN.

In this paper, we propose a blockchain-based federated learning architecture in which a dual-chain structure ensures the secure sharing of topological and model information. In addition, we design a node security evaluation model to cope with malicious nodes' behavior. Furthermore, we improve the traditional PBFT algorithm and propose an enhanced practical Byzantine fault tolerance algorithm. After that, we model the traffic offloading problem as an MDP and take a federated reinforcement learning approach to make dynamic decisions. The main contributions of this paper are summarized as follows:

- To realize safely share network information between nodes, we introduce blockchain technology. Then, we introduce federated learning technology to protect data privacy and reduce the overhead of transmitting raw data in SAGIN. In addition, We present a novel blockchain-based federated learning architecture. Based on the above framework, we propose a dual-blockchain structure, where the topology chain is used to share network topology information securely, and the model chain is used to share the models in the federated learning process securely.
- To solve the problem of secure node selection in traffic offloading, we design a node security evaluation mechanism. During the transmission of packets, the offloading decision is made based on the results of node trusted evaluation, which can effectively deal with the malicious behaviors of malicious nodes.
- We improve the traditional PBFT algorithm based on the node security evaluation mechanism. We first propose a consensus node selection mechanism, after which we further propose an enhanced practical Byzantine fault tolerance (EPBFT) algorithm based on the above mechanism.
- We abstract the blockchain-based traffic offloading problem in SAGIN as an MDP and propose to use an RL-based approach to solve this problem. By combining federated learning techniques, we collect local information to learn the network dynamics, which can effectively reduce the network topology complexity and reduce the communication overhead caused by the transmitted data. In addition, we propose using a model-free A3C algorithm to dynamically learn the network topology and make optimization decisions to minimize the total system delay.

The rest of this paper is organized as follows. In Section II, we describe the related work. Section III presents the system structure, and Section IV presents the system model and problem formulation. In section V, we describe the node security evaluation and consensus mechanism, and blockchain system delay model. In Section VI, we describe the traffic offloading problem, followed by a federated reinforcement learning-based offloading scheme. Section VII designs experiments for evaluating our proposed approach and concludes the whole paper in Section VIII.

II. Related Work

In this section, we introduce some traffic offloading work and blockchain applications in SAGIN. Then, the related work on offloading of federated learning in wireless networks is introduced. In addition, there are also some work introductions on offloading solutions based on MDP and deep reinforcement learning in SAGIN. Finally, the novel federated reinforcement learning technology applied in the wireless network field is introduced.

A. Offloading Solutions in SAGIN

In recent years, some researchers have considered satellites or UAVs in SAGIN as offloading points in traffic offloading work. For example, in the satellite-terrestrial network, Du et al. [19] proposed an auction mechanism for SDN-based traffic offloading, which enables cooperation and competition among beam groups of the system. Considering energy constraints, Li et al. [20] proposed a transmission scheme to enable an energy-efficient RAN by offloading traffic from base stations through satellite's broadcast transmission.

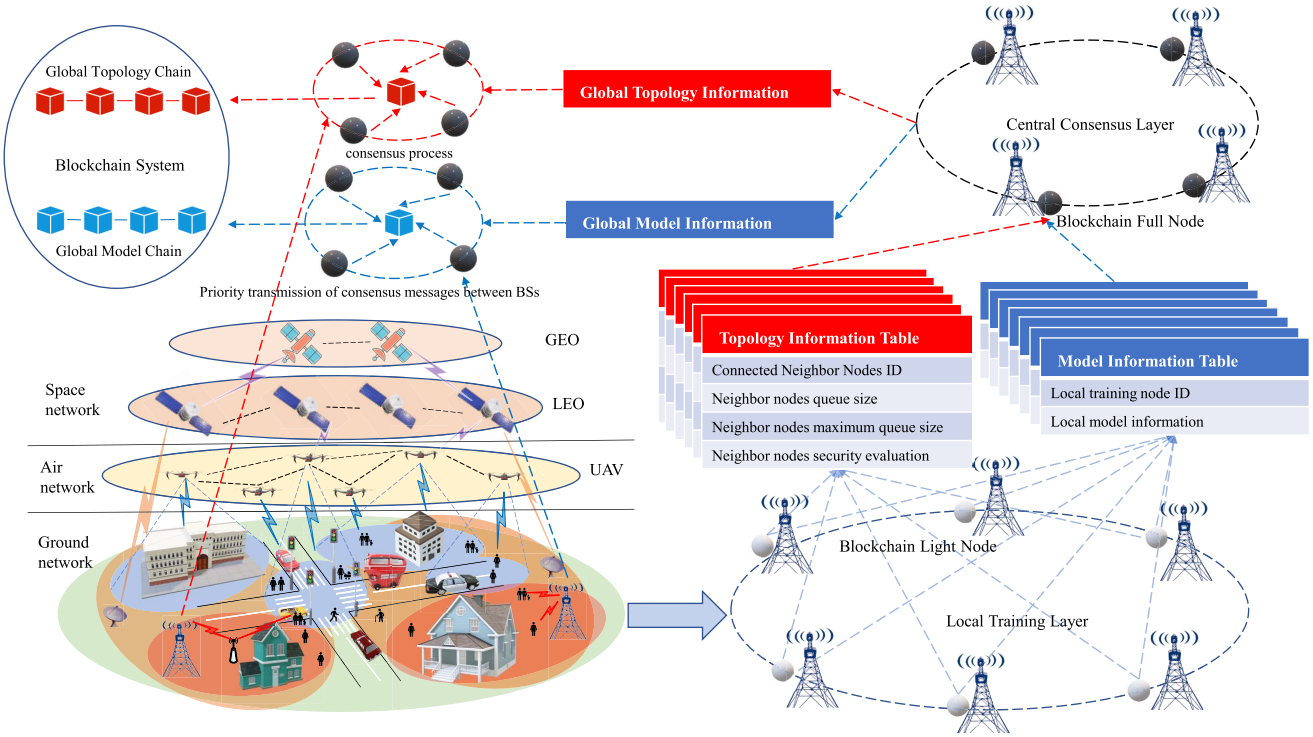


Fig. 1. The architecture of blockchain-based federated learning in SAGIN.

In addition, considering that UAV communication is a promising technology, the work of [21] investigated the UAV-enabled traffic offloading problem in air-ground integrated networks with mixed user traffic. The authors proposed a DNN-based solution to jointly maximize the system load balance and the total UAV reward. In addition, Ng et al. [22] proposed a two-stage stochastic coded offloading scheme in the problem of UAV-assisted edge computing.

The above works only consider satellites and UAVs separately, but the environment is more complex in SAGIN. Mao et al. [23] proposed a deep learning approach to optimizing computation offloading in satellite-UAV-served 6G IoT. Recently, researchers performed traffic offloading work in SAGIN. The authors proposed a reinforcement learning-based traffic offloading method by considering the highly dynamic characteristics of SAGIN [24]. Followed by the work of [25], Zhang et al. proposed an innovative offloading approach that covers the co-existing requirements of two heterogeneous slices of 5G. However, none of these works consider the security issues in a large heterogeneous network such as SAGIN with malicious nodes. At present, there is still a lack of work on traffic offloading in SAGIN. Furthermore, to the best of our knowledge, we were the first to start working on the problem of traffic offloading in SAGIN with malicious nodes, so our current work can be useful for future researchers to use as a reference.

B. The Application of Blockchain in SAGIN

As a widespread technology, blockchain has attracted the attention of many researchers. In the work of [26], authors proposed a collaborative blockchain architecture for SAGIN to achieve secure and efficient management of diverse resources. In addition, in the work of [27], considering the privacy protection in SAGIN cannot meet location-based service (LBS) security requirements well, the authors presented a blockchain-based LBS security preserving trust model and proposed a trust management algorithm that can detect malicious behaviors. In the work of [28], the authors proposed blockchain-based multi-party cooperation and resource-sharing framework and discussed the framework based on the consortium blockchain. Furthermore, some researchers discussed blockchain-based solutions for space-air-ground IoT (SAG-IoT) security and the critical challenges when integrating blockchain in SAG-IoT security services [29]. However, most of the above work is related to resource allocation. So far, no researchers have combined blockchain in SAGIN to do traffic offloading work.

C. Federated Learning for Offloading in Wireless Network

As a distributed machine learning technology, federated learning has been researched on offloading schemes in wireless networks. In the work of [30], to optimize the vehicular network performance in terms of latency and energy consumption, authors proposed an FL-inspired distributed learning framework for computation offloading problems and proposed an evolutionary Genetic Algorithm to solve it. In addition, in the work of [31], the authors studied the fine-grained tactical task offloading mechanism while leveraging the FL process and investigated the dual-interactive bargaining process based on the interdependent relationship between IoT devices and the tactical edge server. Moreover, Zhu et al. [32] introduced a novel prediction-assisted task offloading scheme for power

grid IoT, and they designed a federated learning approach to train the task prediction model.

However, among the above works, the offloading schemes based on federated learning do not consider large dynamic network scenarios. In SAGIN, to make optimal offloading decisions, the real-time dynamics of the network must be considered. A federated learning-based framework that adopts a more intelligent learning approach for offloading decisions is a potential solution.

D. Offloading Solutions Based on MDP and Deep Reinforcement Learning in SAGIN

Due to the high mobility of UAVs and satellites in SAGIN, the communication link is unstable. In addition, due to the highly dynamic network state, using fixed rule-based offloading schemes, such as greedy algorithms, cannot make optimal offloading decisions. Therefore, the offloading decision problem is often described as an MDP problem. By defining the elements in the MDP, deep reinforcement learning (DRL) is used to learn the optimal offloading policy.

Chen et al. [33] first studied the computing offloading problem in SAGIN by modeling the computing offloading decision problem as MDP and then using an online DRL method to learn the optimal offloading strategy dynamically. However, the above schemes need to collect global information to make centralized offloading decisions, which will bring additional energy overhead. Instead, Zhou et al. [34] formulated the task offloading problem as a constrained MDP and employed linear programming to find a stochastic policy. The proposed solution is unsuitable for large-scale network environments and may not perform optimally in the case of multiple UAVs and satellites in SAGIN. Wang et al. [35] formulated the task offloading problem as MDP and proposed a CL-MADDPG method to learn offloading strategy. However, the proposed scheme relies on the predetermined flight trajectory of UAVs and is unsuitable for multi-UAV situations. By summarizing the above work, it is feasible to describe the traffic offloading decision problem in SAGIN as MDP and use DRL to solve it.

E. Federated Reinforcement Learning in Wireless Network

Federated Reinforcement Learning (FRL) is a relatively new technology in wireless networks. To solve the user access control problem, Cao et al. [36] proposed an intelligent user access control scheme and combined federated learning and deep Q-network (DQN) to deal with the problem of decision-making. However, the proposed scheme may not suit a highly dynamic network such as SAGIN. In addition, FRL also plays a vital role in edge caching. Zhang et al. [37] studied the cooperative edge caching problem in fog wireless access networks. Considering the NP-hard nature of the problem, the authors proposed a cooperative edge caching scheme of federated learning combined with a dueling deep Q-network. FRL also has related research in the field of resource allocation. In [38], the authors proposed an FRL-based channel resource allocation framework for 5G/B5G networks, which achieved good results in throughput. In addition, the authors highlighted six potential applications of the framework. In the above work, FRL technology has achieved superior performance. However, due to the highly dynamic and heterogeneous nature of SAGIN, it is a challenge to design traffic offloading schemes using FRL technology.

III. System Structure

A. Blockchain-Based Federated Learning Architecture in SAGIN

Fig. 1 shows the blockchain-based federated learning framework in SAGIN, which is divided into two layers, i.e., the local training layer and the central consensus layer.

1) Local Training Layer: This layer consists of the BSs of the terrestrial segment in SAGIN. The connected UEs and BSs carry out the packet transmission in the terrestrial segment. The system enables the offloading scheme when the ground BS exceeds the load. The local training node collects the topology information of the neighbor nodes in real-time and transmits it to the central consensus layer, where the topology information mainly includes the node's connectivity with neighbor nodes and the node security evaluation information. We use the selected BSs as local training nodes for federated learning and interact with the central consensus layer for model parameters.
2) Central Consensus Layer: The nodes of the central consensus layer are selected by a selection mechanism (Section V.B) based on the node security evaluation. We select n BS nodes with the highest security evaluation values as the central consensus layer nodes. The specific selection mechanism is shown in Algorithm 1. This layer collects topology information and local model parameters uploaded by the local training layer. And it evaluates the training results of each local training node as part of the security evaluation. Furthermore, the central consensus layer operates as a distributed blockchain. The nodes in this layer can share model parameters with other nodes transactionally through the blockchain. This approach effectively solves the problem of poor training results due to attacks on the central control node under the federated learning framework.
3) Blockchain System: In this paper, we propose a dual-chain structure for consortium blockchain, in which the topology chain (red blockchain) stores network topology information, and the model chain (blue blockchain) stores the model information. We use the local training nodes as the light nodes of the blockchain to collect topology information and cache block header and traffic offloading related information. In addition, we use the central consensus nodes as the full nodes, which need to store all relevant information and participate in the consensus process. This setup facilitates the querying for the subsequent traffic offloading process.

B. Traffic Offloading Based on Double Blockchain

1) Network Topology Information Chain (Topology Chain): The local training node collects the network topology

information of the neighboring nodes to form a transaction record to send to the central consensus layer. The network topology information consists of the one-hop and two-hop BS sets, UAV sets, LEO sets, GEO sets, queue resources of the nodes, and node security evaluation information (described in Section V.A) that the local training node can connect. The central consensus layer will package this information into a block and add it to the blockchain after going through the consensus process. During the traffic offloading process, the network topology information stored through the blockchain can effectively ensure security and plays an essential role in the model validation process afterward.

2) Federated Learning Model Information Chain (Model Chain): After local training, local training nodes upload local models to the central consensus layer. The central consensus layer node first aggregates the local models, then packages the latest global model information into blocks and adds them to the blockchain after passing the consensus process. Furthermore, the central consensus layer validates the local model and uses it as part of the security evaluation based on its performance. After that, local training nodes can get the global model directly through the blockchain, which reduces the interaction between the local training layer and the central consensus layer, and ensures the security of obtaining the global model.

IV. System Model and Problem Formulation

A. Network Model

As shown in Fig. 1, we consider a SAGIN network architecture that consists of three network segments, i.e., the ground segment, the air segment, and the space segment. The user equipment (UE) and terrestrial BS belong to the ground segment, which has limited coverage and capacity. The UEs run applications that generate source data for transmission. In the air segment, the UAV is highly mobile and flexible as a flying base station to provide network access to the ground segment. These UAVs are set to fly at a fixed speed and a mobility model. In the space segment, several LEOs provide intermittent coverage of the target area, and one GEO provides full coverage.

We model SAGIN as a weighted directed graph $G = (N, E)$, where $N = \{UN, RN\}$ denotes all nodes in the network. We assume that there are z UE nodes $UN = \{un_1, un_2, \ldots, un_z\}$, which generate packets and transmit packets to other nodes. Moreover, $RN = \{B, U, SA\}$ denotes the relay nodes in SAGIN, where $B = \{b_1, b_2, \ldots, b_b\}$ denotes the BS set, $U = \{u_1, u_2, \ldots u_u\}$ denotes the set of UAVs and $SA = \{g, l_1, l_2, \ldots, l_s\}$ denotes the set of satellites, specifically, g represents GEO and l_i represents LEO. The weighted edge $E = \{e_{01}, e_{10}, \ldots, e_{xx}\}$ denotes the link state between two nodes in the network, where if $e_{ij} = 1$ then the two nodes have established communication, and if $e_{ij} = 0$ then the two nodes have not established communication. At time t, each UE generates normally distributed packets of size $sz \in (\mu, \omega^2)$ and a defined routing path $path = \{un_s \longrightarrow un_d\}$ sent to device $un_d \in UN$. It can be further expressed as $P_{un_s \longrightarrow un_d} = \{sz_{path}, path, T_d\}$, where T_d denotes the average transmission delay of the packet.

B. Communication Model

For UAV-ground communication, we consider the path loss model between UAV and BS in [39]. The path loss between UAV and BS can be calculated as follows:

$$L(\varsigma, \phi) = 10\varphi \log(\varsigma) + \xi(\phi - \phi_0) e^{\frac{\phi_0 - \phi}{h}} + k_0, \tag{1}$$

where ς denotes the horizontal distance between UAV and BS, ϕ denotes the longitudinal angle between UAV and BS, φ denotes the terrestrial path loss exponent, ϕ_0 denotes the angle offset, ξ denotes the excess path loss scalar, h denotes the angle scalar, and k_0 denotes the excess path loss offset [39]. Since the UAV moves according to the designed movement model, the parameters between UAV and BS change continuously. Therefore, based on the above formula, we can calculate the transmission rate between UAV and BS at time slot t according to the formula as follows:

$$tr_t^{UB} = W_{UB} \log_2 \left(1 + \frac{P_{UB} \cdot 10^{-\frac{L}{10}}}{\sigma_{UB}^2}\right), \tag{2}$$

where W_{UB} denotes the channel bandwidth between UAV and BS, P_{UB} denotes the transmission power between UAV and BS, σ_{UB}^2 denotes the noise power between UAV and BS, L denotes the average path loss between UAV and BS.

In addition, in the process of offloading traffic to the satellite, we consider the impact of rain attenuation on the signal. We regard the rain attenuation during satellite and terrestrial communication as a random process based on Weibull [40]. Due to the distance change caused by the trajectory movement of the UAV is insignificant compared to the distance between the satellite and the UAV, we assume that the movement of the UAV is stationary relative to the channel. We can calculate the power attenuation as follows:

$$I_{US}^2 = \frac{G_{US} G_{SU} \lambda_{US}^2}{(4\pi d_{US})^2} 10^{-\frac{F_{rain}}{10}}, \tag{3}$$

where G_{US} and G_{SU} are the antenna gains of the UAV and the satellite, respectively, λ_{US} is the wavelength, and d_{US} is the distance between the satellite and the UAV. Rain attenuation F_{rain} is modeled by Weibull distribution [40]. Based on equation (3), we can calculate the transmission rate between satellite and UAV as follows:

$$tr_t^{US} = W_{US} \log_2 \left(1 + \frac{P_{US} \cdot I_{US}^2}{\sigma_{US}^2}\right), \tag{4}$$

where W_{US} denotes the channel bandwidth of the UAV and the satellite communication link, P_{US} denotes the transmission power between UAV and satellite, and σ_{US}^2 denotes the power of the background noise.

C. Blockchain Latency Model

First, each network node of SAGIN is certified by the certification authority to maintain the alliance blockchain. In this

scenario, due to the high capacity and stability of the BSs in the ground segment, we select some BSs as full nodes for storing all the information and participating in the consensus process (selection is made based on the results of the node security evaluation). In the blockchain, only full nodes can participate in the consensus process. In addition, due to the communication instability caused by the high mobility of UAV and LEO, we consider them as light nodes for storing partial information without participating in the consensus process. GEO is used as a light node because of its relatively high transmission delay due to factors such as rain attenuation.

In this paper, we propose an enhanced practical byzantine fault tolerance algorithm, and the consensus process is described in Section V.B.

The main compositions of delay in blockchain system include consensus delay $T_{con}(t)$ and node updating delay $T_{upd}(t)$. For every M times of consensus process, the system launches a node updating operation. Hence we can express the average delay of the blockchain system as follows:

$$T_c^b(t) = \frac{MT_{con}(t) + T_{upd}(t)}{M}, \tag{5}$$

The detailed calculation and analysis are in section V.C.

In this paper, we perform traffic offloading work in a SAGIN environment to reduce the latency of the entire network environment and thus ensure the quality of service for users. Therefore, to ensure a safe traffic offloading work in a SAGIN environment where malicious nodes exist, we want to impose a limit on the consensus latency of the blockchain [41]:

$$\forall_t T_c^b(t) \leq \left(T_c^b\right)_{\min}, \tag{6}$$

where $(T_c^b)_{\min}$ is the delay limit to ensure the safety of the system.

D. Traffic Offloading Delay Model

During the process of traffic offloading, the end-to-end delay is the sum of all delay components on the path from the source node to the destination node, which is divided into two components: delay on the link and queuing delay. In addition, the delay on the link is divided into propagation delay and transmission delay. Therefore, the delay of one-hop between node i and node j can be expressed as follows:

$$T_{path_{i,j}} = \frac{x_{i,j}}{\iota} + \frac{\ell}{tr_t^{i,j}} + \eta\frac{\ell}{tr_t^{i,j}}, \tag{7}$$

where $x_{i,j}$ denotes the distance between node i and node j, ι denotes the signal transmission speed, ℓ denotes the length of the transmission packet, $tr_t^{i,j}$ denotes the transmission rate between node i and node j, and η denotes the amount of queued packets [24]. Based on the equation obtained above, we can calculate that the total end-to-end delay can be expressed in the following form for the case of $path$ determination [24]:

$$\mathcal{T}(path) = \sum_{path_{i,j} \in path} T_{path_{i,j}}, \tag{8}$$

where $path_{i,j}$ represents the one-hop path between node i and node j.

E. Problem Formulation

In this paper, we study the problem of trusted traffic offloading in the presence of malicious nodes in SAGIN. The optimization goal is to minimize the total delay of the network nodes in SAGIN under the security constraint by optimizing the routing paths. Specifically, we describe the problem as an objective function that minimizes the delay of packets in the overall network under the security constraint (6) by generating optimized routing paths $P_{un_s \to un_d}$ within time slot t by adopting a traffic offloading approach $F_{\rho,h}(un_s \to un_d, Path)$. Among them, $Path = \{path_1, \ldots, path_Y\}$ is the candidate offloading path set, and $path = \{RN \to un_d\}$ represents the path from the relay node $RN = \{B, U, SA\}$ to the destination node un_d. Furthermore, the problem depends on two explicit parameters ρ and y, where $\rho \in \{0, 1\}$ denotes whether to offload or not, and y denotes the y-th path. Therefore, the objective function formula of this paper is as follows [24]:

$$\begin{aligned} &\arg\min_{\rho,y} \sum_T \sum_N \mathcal{T}\left(F_{\rho,y}\left(un_s \to un_d, Path\right)\right), \\ &\text{s.t.} \quad C_1 : \forall_{e_{ij}} \sum_N sz_{path} \leq tr_t^{i,j}, sz_{path} \in P_{un_s \to un_d}, \\ &\qquad C_2 : un_s, un_d \in UN, \\ &\qquad C_3 : path_y \in Path, RN \neq un_s, RN \cap FN = \varnothing, \\ &\qquad C_4 : (6). \end{aligned} \tag{9}$$

where $\mathcal{T}(\cdot)$ represents the total delay of the calculated packet in the path, and FN represents malicious node set.

V. Proposed Node Security Evaluation and Consensus Mechanism

When malicious nodes exist in SAGIN, misinformation (e.g., topology and model information) transmitted by malicious nodes can cause traffic to fail to reach its destination. As a result, the malicious behavior of malicious nodes can lead to performance degradation such as delay and throughput of the system. To minimize the harm of malicious nodes, we propose a node security evaluation and consensus mechanism. We first analyze the behavior of malicious nodes and then perform security evaluations based on different behaviors. In addition, we design an enhanced practical byzantine fault tolerance algorithm to secure the blockchain system and thus ensure information security during traffic offloading. Finally, we analyze blockchain system delay.

A. Node Security Evaluation Mechanism

To ensure the security of the node, we consider the security evaluation of the nodes. In this paper, we assume that the malicious nodes in SAGIN have the following three malicious behaviors.

- Malicious packet dropping behavior: When a malicious node receives a packet, it drops the packet with a certain probability.
- Malicious packet transmission behavior: When a malicious node receives a packet, the malicious node does not transmit the packet according to the specified routing path or the offloading scheme path. For example, a malicious

node arbitrarily transmits packets to a neighbor node that has established a link or maliciously deviates from the defined path for long-distance transmission.

- Malicious model upload behavior: Suppose the malicious node is a training node for federated learning. In that case, the malicious node uploads the poorly performing training model to the central consensus layer.

For the three malicious behaviors mentioned above, we propose a security value model that combines the three malicious behaviors evaluated.

1) Node delivery evaluation: In the network, each node, excluding the source and destination nodes, receives packets from other nodes and transmits packets to the next-hop node. Malicious nodes generally avoid receiving and transmitting packets according to the normal process, so we calculate the node's delivery rate as a partial basis for evaluating whether a node is a malicious node. In time slot t, node N_i records information about the packets transmitted and received, including packets received from neighboring nodes and packets sent to neighboring nodes. The local training node collects and stores this information in the blockchain as part of the security evaluation. We evaluate the delivery trust value of node N_i by examining the delivery rate of packets from this node. Thus, the evaluation value of node N_i in packet delivery rate in time slot t is calculated as follows:

$$\mathcal{O}_t^{N_i} = \frac{C_t^{N_i \to RN_{nei}(j)}}{C_t^{RN_{nei}(k) \to N_i}}, \tag{10}$$

where $C_t^{N_i \to RN_{nei}(j)}$ denotes the number of packets received by the neighboring nodes $RN_{nei}(j)$ of node N_i from N_i in time slot t, $C_t^{RN_{nei}(k) \to N_i}$ denotes the number of packets transmitted by the neighboring nodes $RN_{nei}(k)$ to node N_i in time slot t.

2) Node transmission path evaluation: Malicious nodes usually do not follow the specified path for transmitting packets to the next-hop node, which leads to a significant increase in packet transmission delay. We calculate the correct rate of a node's transmission path as part of the basis for assessing whether the node is malicious. When a local training node receives a packet, it checks whether the last-hop node of the packet has taken the optimal path for transmission and records it through the topology information chain. In time slot t, the local training node will upload the recorded information of the relevant node into the topology information chain. When the node is not recorded, we default the node is safe. If there are detection errors, the transmission path evaluation value will be normalized as the total number of detections increases. We can calculate the evaluation value of node N_i in terms of the transmission path in time slot t by the following formula:

$$\mathcal{P}_t^{N_i} = 1 - \frac{\mathfrak{D}_t^{N_i}}{1 + \mathbb{D}_t^{N_i}}, \tag{11}$$

where $\mathfrak{D}_t^{N_i}$ denotes the number of times that the next hop transmitted by node N_i is detected as not the optimal path in time slot t, $\mathbb{D}_t^{N_i}$ denotes the total number of times that node N_i is detected in time slot t.

3) Federated learning model evaluation: Malicious nodes usually upload models that do not work well to the central aggregation point. In this paper, local training nodes upload models to the central consensus layer. We partly evaluate whether the local training node is a malicious node by testing the performance of the uploaded models. After completing one training iteration, the local training node will upload the local model to the central consensus layer, which will validate the model based on the information in the topology chain. We can judge the accuracy of the local model by validation. We use the following formula to calculate the security evaluation value of the federated learning nodes N_i in terms of models in time slot t:

$$\mathcal{M}_t^{N_i} = 1 - \frac{\mathfrak{Z}_t^{N_i}}{1 + \mathbb{Z}_t^{N_i}}, \tag{12}$$

where $\mathfrak{Z}_t^{N_i}$ denotes the number of times that node N_i is detected to upload the inaccurate model in time slot t, and $\mathbb{Z}_t^{N_i}$ denotes the total number of times that node N_i is detected in time slot t.

4) Comprehensive evaluation: After evaluating the above three malicious behaviors, the comprehensive evaluation value calculation formula is as follows:

$$\mathcal{E}_t^{N_i} = \mathcal{O}_t^{N_i} \mathcal{P}_t^{N_i} \mathcal{M}_t^{N_i}, \tag{13}$$

Among them, since the evaluation value range of the three malicious behaviors belongs to $(0, 1]$, the range of the comprehensive security evaluation value belongs to $(0, 1]$.

B. Consensus Mechanism

Practical Byzantine Fault Tolerance (PBFT) is a form of state machine replication [42]. We assume that there are f malicious nodes in the network and that the system is secure when the number of summary points $\mathbb{N} \geq 3f + 1$. In the PBFT algorithm, there is only one primary node in a view, and all others are non-primary nodes. In the traditional PBFT implementation of consensus, the primary node is selected by the formula $P = v \; mod \; \mathbb{N}$, where v is the view number. When the primary node needs to be replaced, the view is changed and a new primary node is selected according to the above formula. However, this selection of primary nodes by the residual method is arbitrary and uncontrollable, and the security of the selected primary nodes cannot be guaranteed.

In this paper, a security-based enhanced Practical Byzantine Fault Tolerance (EPBFT) algorithm is designed to implement the consensus process. We propose a central consensus layer node selection mechanism based on the node security evaluation to ensure the security of consensus nodes. Algorithm 1 describes the selection mechanism of the central consensus layer. We select some BS nodes with security evaluation value greater than the security threshold as the central consensus

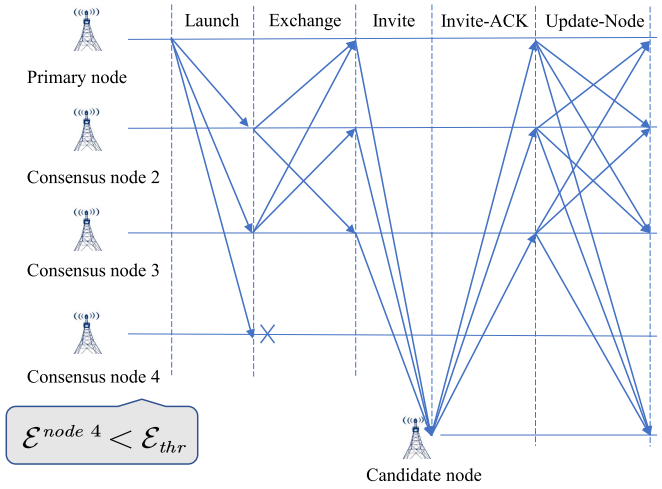


Fig. 2. The process of updating consensus nodes in EPBFT.

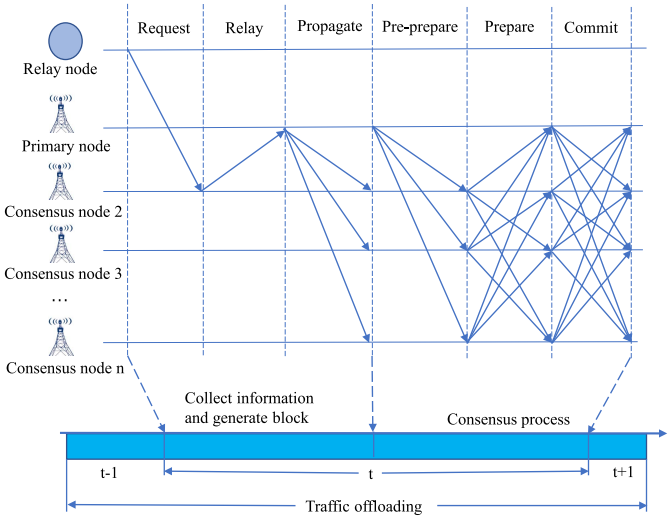


Fig. 3. The traffic offloading and EPBFT consensus process.

Algorithm 1 Central Consensus Layer Selection Mechanism Algorithm

Input: BS set B, central consensus node list $Clist$, security threshold $\mathcal{E}_{thr}$.
Output: $Clist$.
Initialize $Clist$.
while running **do**
  if M rounds of consensus process finished **then**
    while $min(\mathcal{E}^{B_i}) < \mathcal{E}_{thr}$ **do**
      Delete node B_i from $Clist$.
      Add candidate node B_j with $max(\mathcal{E}^{B_j})$ to $Clist$ as Fig. 2.
    end while
  end if
end while

layer nodes. In order to better maintain the alliance chain, we add the function of dynamically adding and deleting nodes to PBFT. Among them, the process of dynamically updating nodes in Algorithm 1 is shown in Fig. 2. The specific process of updating consensus nodes is shown as follows:

1) Launch: After M times of the consensus process, the primary node launches a node updating process and sends the consensus node with the lowest reputation and security value to other consensus nodes.
2) Exchange: After consensus nodes receive the LAUNCH message from the primary node, each node sends the consensus node with the lowest reputation and security value to other consensus nodes. When a consensus node receives $2f+1$ messages (including in the Launch phase) that the consensus node with the lowest reputation is below the threshold, it will be deleted from its consensus list.
3) Invite: If the consensus node deleted the node with a low reputation in the Exchange phase, it sends an INVITE message to a non-consensus node with the highest security value.
4) Invite-ACK: When the candidate consensus node receives $2f + 1$ invitation messages, it sends a reply message Invite-ACK to the central consensus node that sent the invitation.
5) Update-Node: When the central consensus node receives $2f+1$ Invite-ACK messages, the central consensus node sends update node information Update-Node to other central consensus nodes (including the candidate consensus node.) When the central consensus node receives $2f + 1$ Update-Node messages, the candidate central consensus node becomes the formal central consensus node.

In addition, we select the node with the highest security value from the central consensus layer as the primary node. When a primary node breaks down, we re-select the primary node from the central consensus layer node list. Fig. 3 shows the traffic offloading and the consensus process of EPBFT. As shown in Fig. 3, EPBFT consists of a six-stage process, which are Request, Relay, Propagate, Pre-prepare, Prepare, and Commit, and are analyzed as follows:

1) Request: Relay node sends its own information to the nearest surrounding consensus node.
2) Relay: After receiving the information from the surrounding relay node, the consensus node packages it and forwards it to the primary node.
3) Propagate: When the primary node receives the message from the consensus node, it will broadcast the message to other consensus nodes. After the primary node verifies the message authentication code (MAC), it will open the three-stage protocol, i.e., Pre-prepare, Prepare, and Commit.
4) Pre-prepare: In this phase, the primary node first packages the validated transactions into a block. Then the primary node will broadcast a PRE-PREPARE message to all non-primary nodes. The PRE-PREPARE message format is $\langle\langle$ PRE-PREPARE, $v, n, d\rangle_{\sigma_p}, m\rangle$, where v indicates the view in which the message is being sent,

n is the request sequence number, m is the request message, and d is m's digest [42]. For non-primary nodes to verify PRE-PREPARE messages, the primary node also generates a signature and MACs.

5) Prepare: In this phase, each non-primary node receives a PRE-PREPARE message and verifies the signatures and MACs. Then the non-primary np_i then broadcasts a PREPARE$\langle$ PREPARE, $v, n, d, np_i\rangle_{\sigma_i}$ message to all other nodes, while they receive PREPARE messages from other nodes. Furthermore, each non-primary node receives the PREPARE message and compares it with PRE-PREPARE. We know that in the PBFT protocol, when the full node receives a valid signature greater than or equal to $2f$, then the relevant message is considered correct. If the verification passes, these non-primary nodes add an additional signature message and MACs.
6) Commit: In this phase, nodes that receive more than two PREPARE messages broadcast COMMIT messages to other nodes for verification. If the number of COMMIT messages received by the node reaches $2f + 1$, then consensus is reached among the consensus nodes.

In addition, in order to reach the consensus process faster, we set the consensus messages between BSs to have high priority, which further enhances the security of the traffic offloading process.

C. Blockchain System Delay

This section calculates the latency of the blockchain system by analyzing the cost of the blockchain system. The propagation delay can be negligible because the consensus process occurs between BS nodes. In this paper, as shown in Fig. 1, we consider blockchain signaling to have the highest priority and therefore do not consider queuing delays.

We assume the number of consensus nodes is N, and the number of malicious nodes is f. According to [43], we consider only the computational costs of the cryptographic operations, including verifying signatures, generating MACs, and verifying MACs, with costs of C_s, C_m, C_v cycles respectively. The processing speed of relay nodes is $\boldsymbol{k}$ GHz. The cost of blockchain system is divided into two parts: consensus cost and node updating cost.

1) Consensus cost: In the information collection phase, the delay is a fixed value T_g set by the system. The consensus cost consists of three phases: Pre-prepare, Prepare and Commit. In Pre-prepare phase, the primary node sends PRE-PREPARE messages to other $N-1$ nodes, and other nodes verify the signature of the primary node and the MAC of the message. Furthermore, in Prepare phase, consensus nodes send PREPARE messages to other $N-1$ nodes and verify $2f$ signatures and MACs. In Commit phase, every node sends a COMMIT message to other $N-1$ nodes and verifies $2f+1$ signatures and MACs. Therefore, the consensus cost of the primary node and consensus node can be expressed as follows:

$$C_p^{con} = 2(N-1)C_m + (4f+1) \times (C_s + C_v), \quad (14)$$
$$C_r^{con} = (2N-1)C_m + (4f+1) \times C_s + 2(2f+1)C_v. \quad (15)$$

The consensus delay can be calculated as follows:

$$T_{con}(t) = \frac{MAX(C_p^{con}, C_r^{con})}{\boldsymbol{k}} + T_g. \quad (16)$$

2) Node updating cost: We consider the blockchain system in a stable state (i.e., No node with a security value below the threshold in the consensus node), so we only consider the Launch phase and Exchange phase. In the launch phase, to check and update information, the primary node sends its information of security value to other consensus nodes, adds the node's signature in the message, and generates $N-1$ MACs. In the Exchange phase, the nodes exchange information with the highest and lowest reputation. The consensus node generates $N-1$ MACs, and each node receives and verifies the MAC and signature of $2f$ messages. Therefore, we can calculate the node updating cost of the primary node and consensus node as follows:

$$C_p^{upd} = (N-1)C_m + 2f \times (C_s + C_v), \quad (17)$$
$$C_r^{upd} = N \times C_m + 2f \times C_s + 2(f+1)C_v. \quad (18)$$

The node updating delay can be express as follows:

$$T_{upd}(t) = \frac{MAX(C_p^{upd}, C_r^{upd})}{\boldsymbol{k}}. \quad (19)$$

According to equation (5), the average blockchain system average latency is:

$$T_c^b(t) = \frac{M(MAX(C_p^{con}, C_r^{con})) + \boldsymbol{k}T_g) + MAX(C_p^{con}, C_r^{con})}{\boldsymbol{k}M}. \quad (20)$$

VI. Traffic Offloading Problem Formulation and Offloading Algorithm Based on Federated Reinforcement Learning

We designed an online traffic offloading method for this system. For each time slot t, packets will be transmitted according to the initially designed routing path, or the packets will be offloaded to BS, UAV, or satellite through the offloading algorithm. The goal is to reduce the total transmission delay and thus reduce the cost. However, it is challenging to make the offloading decisions to realize this goal. In SAGIN's highly dynamic network environment, the fixed offloading decision is not accurate, and it is necessary to analyze the current network environment to make further offloading decisions. Therefore, we can abstract the complex traffic offloading decision problem into an MDP.

A. Markov Decision Process

An MDP problem consists of five parts: the state space denoted by S, action space denoted by A, the transition probability denoted by P, the reward function denoted by r, and the policy denoted by π. The MDP of the traffic offloading problem is defined as follows:

- State space S: The state space is defined as a multi-dimensional array that contains all the one-hop

and two-hop neighbor-related information around the node. This coincides with the federated learning we will consider later. We do not need to know the global network state, only the state information around the node. This information includes dynamic and static information, which allows us to parse the state of neighbor nodes around one hop and two hops. According to the different characteristics of different devices, some static parameters can be arbitrarily set. In this paper, we use the Hello protocol to capture the relevant information of adjacent one-hop and two-hop nodes. Then the relevant information is stored in the blockchain system to ensure the security of the information.

$$S_{RN_i}(t) = \left\{ Q_t^{RN_{nei}(RN_i)}, Q_t^{RN_{nei}(RN_{nei}(RN_i))} \right\}, \tag{21}$$

where Q_t^* represents relevant information of node, including basic information of offloading decisions, such as node security evaluation value $\mathcal{E}_t^{RN_i}$, node maximum queue size, node current queue size. And $RN_{nei}(\cdot)$ represents the set of neighbor node.

- Action space A: The action space is defined as follows:

$$a(t) = \{0, RN_i\}, RN_i \in path_y, \tag{22}$$

where the first element denotes that the system uses the originally set routing path to transmit the packet to the next node, and where the second element denotes the offloading of packets to neighbor nodes, such as BSs, UAVs and satellites, according to the new routing path $path_y$ generated by the offloading decision.

- Transition probability P: The transition probability $P(s_{t+1} \mid s_t, a_t)$ is difficult to model accurately. This is because of the uncertainty brought by the dynamics of the UAV. If the packages are to be offloaded to the UAV, in the process of transmission, the UAV may fly out of the communication range. Whether packets can be transmitted to the UAV depends on the UAV's mobility model, bandwidth, and other factors, which are highly dynamic.
- Reward function r: To minimize the total delay of the system, we set the reward function as follows:

$$r(\mathrm{s_t}, a_\mathrm{t}) = \begin{cases} \dfrac{1}{\mathcal{T}_{P_{un_s \to un_d}}}, & arrive \\ -\mathcal{T}_{P_{un_s \to un_d}}, & drop \end{cases}, \tag{23}$$

where $\mathcal{T}_{P_{un_s \to un_d}}$ denotes end-to-end delay of packet. When the data packet arrives at the destination node normally, take the reverse of the delay; when the packet is dropped, take the negative of delay.

Therefore, the value function is expressed as the long-term discounted reward under the policy π when the system state starts from s, as follows:

$$V(s \mid \pi) = E\left[\sum_{t=0}^{\infty} \gamma^t r(\mathrm{s_t}, a_\mathrm{t}) \mid s_0 = s, \pi \right], \tag{24}$$

where $\gamma \in [0, 1]$ is a discount factor, and the expectation is to take all possible state trajectories from the initial state s.

- Policy π: The online traffic offloading method is to maximize the value function of each state with parameter π':

$$\pi'(s) = \underset{a}{\operatorname{argmax}} \sum_{s'} P(s' \mid s, a)[r(s, a) + \gamma V(s' \mid \pi')]. \tag{25}$$

Therefore, the objective function (9) is transformed to find the optimal policy π for choosing action a at state s, which minimizes the package delay.

B. BFA3C-Based Dynamic Traffic Offloading Decision Making

In this paper, we consider a blockchain-based federated learning approach to solve the problem of traffic offloading. We chose a federated approach for several reasons. Firstly, the resource pools in SAGIN are huge, resulting in a high overhead required to manage the resources. If federated learning manages the resource pool, it can avoid the high computational complexity caused by controlling it through a global centralized controller. Secondly, federated learning does not require global information, which can protect users' privacy and prevent information leakage during information transmission. In addition, this method also reduces the communication burden of a highly heterogeneous network like SAGIN.

Since SAGIN contains many highly mobile nodes, it is difficult for us to accurately model the reward function and transition probability in Section VI.A. Therefore, the proposed traffic offloading problem can be solved by model-free RL-based methods, such as the DDQN method [24]. The DDQN method has shown excellent performance in solving the relatively large state space and action space problem in SAGIN. However, due to the large state space, DDQN requires much time to train. Furthermore, it is necessary to sample in the experience playback set during training. Although this eliminates the relevance of the samples, the importance of different samples is different. It is a challenge to combine the research questions and sample methods effectively. In addition, the authors [33] used the actor-critic (AC) algorithm to solve the computational offloading problem in SAGIN. However, the AC algorithm is difficult to converge during the training process. Therefore, in this paper, we propose a Blockchain-based Federated Asynchronous Advantage Actor-Critic (BFA3C) approach to solve the above problem.

The architecture of BFA3C is shown in Fig. 4. The system selects some BSs with high security evaluation values as local training nodes based on the node security evaluation model, thus ensuring the security of local training nodes. BFA3C has a global neural network model (i.e., a central weight matrix) that contains an actor-network and a critic-network, and several local network models (i.e., a local weight matrix) consisting of local actor-network and local critic-network. Each local training node has its local network, which explores the environment separately and periodically feeds

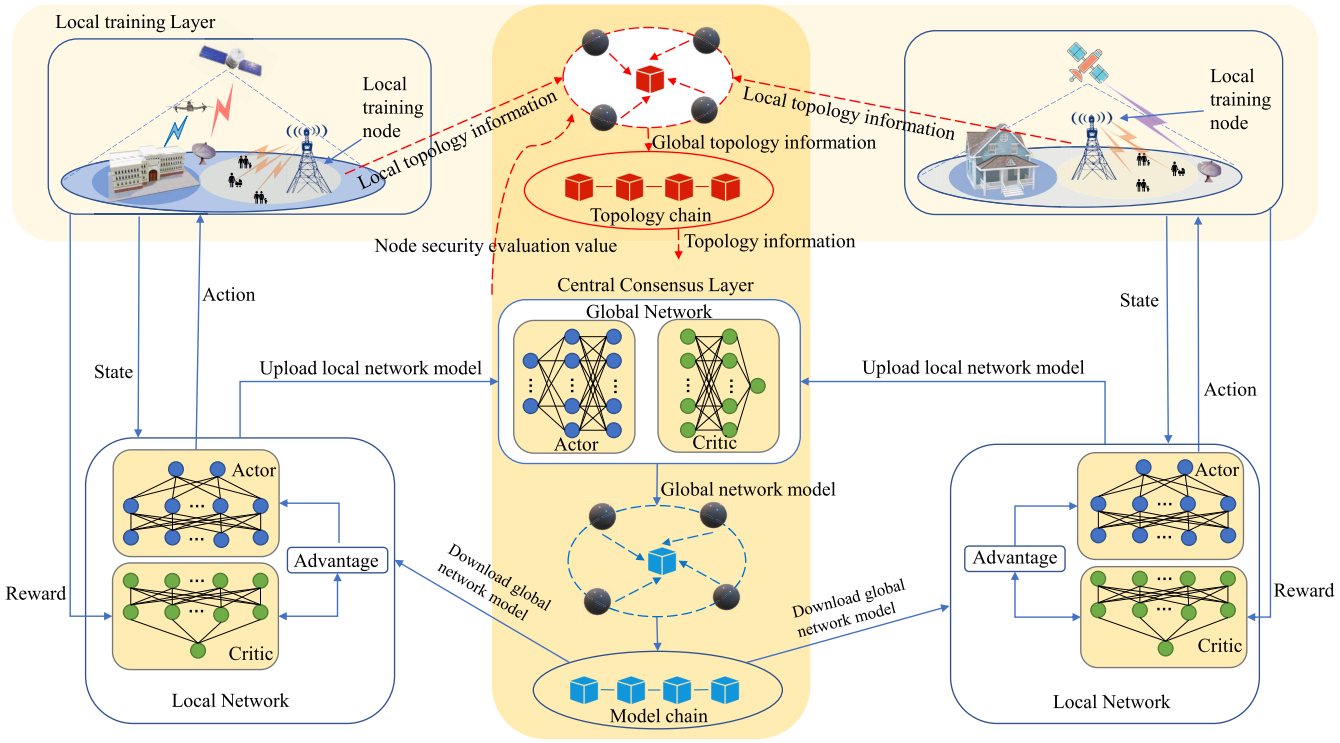


Fig. 4. The architecture of BFA3C-based traffic offloading.

the results of local network learning to the central consensus layer. At the same time, the topology information obtained by the local network is uploaded to the central consensus layer. The topology information is added to the topology chain after the consensus process. The central consensus layer aggregates the uploaded local models and add the global models into the model chain by packing them into a block. Local training nodes obtain the latest global model from the model chain and update the local model. In addition, the central consensus layer obtains topology information from the topology chain as model input to validate the local model, and the validation result will be added to the topology chain to update the node security evaluation information.

In A3C algorithm, the policy is parameterized by the vector θ, i.e., $\pi(a \mid s;\theta)$, the value function is parameterized by the vector θ_v, i.e., $V(s_t;\theta_v)$. In addition, the algorithm uses k-step sampling to make the critic network update the value function closer to the actual reward value. Therefore, the discount accumulated returns is given by

$$R_t = \sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V(s_{t+k};\theta_v), \tag{26}$$

To reduce the variance of the estimation, we define advantage function $A(s_t, a_t; \theta, \theta_v)$ [44] as the evaluation point of critic, the expression of state s_t at time t considering the parameter θ_v as follows:

$$\begin{aligned} A(s_t, a_t; \theta, \theta_v) &= R_t - V(s_t;\theta_v) \\ &= \sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V(s_{t+k};\theta_v) - V(s_t;\theta_v). \end{aligned} \tag{27}$$

Based on the advantage function, the policy loss function is defined as follows:

$$L_\pi(\theta) = \log \pi(a_t \mid s_t;\theta) A(s_t, a_t; \theta, \theta_v) + \alpha Z(\pi(s_t;\theta)). \tag{28}$$

where hyperparameter α controls the strength of the entropy regularization term [44], entropy $Z(\pi(s_t;\theta))$ is used to prevent premature entry into the suboptimal strategy. The corresponding value loss function is defined as follows:

$$L_v(\theta_v) = (R_t - V(s_t;\theta_v))^2. \tag{29}$$

Therefore, the critic-network for gradient update as follows:

$$d\theta_v \leftarrow d\theta_v + \frac{\partial (R_t - V(s_t;\theta_v))^2}{\partial \theta'_v}, \tag{30}$$

The actor-network for gradient update as follows:

$$\begin{aligned} d\theta \leftarrow d\theta + \nabla_{\theta'} \log \pi(a_t \mid s_t;\theta') A(s_t, a_t; \theta, \theta_v) \\ + \alpha \nabla_{\theta'} Z(\pi(s_t, \theta')). \end{aligned} \tag{31}$$

In addition, the global parameters θ and θ_v update asynchronously based on the RMSProp algorithm.

The complete traffic offloading method we proposed is shown in Algorithm 2. And the BFA3C-based traffic offloading algorithm is described as follows. First, the algorithm initializes actor-network and critic-network parameters in the central consensus layer. When the local training nodes start receiving requests for transmission packets, the local training nodes obtain the corresponding pre-trained models from the model chain and perform online learning. The local training node selects an action for offloading decision based on the policy in the local actor-network by collecting information

Algorithm 2 The BFA3C-Based Traffic Offloading Algorithm

Initialize the critic network with parameter θ_v.
Initialize the actor network with parameter θ.
while running **do**
  Reset the gradient of actor network $d\theta \leftarrow 0$.
  Reset the gradient of critic network $d\theta_v \leftarrow 0$.
  Synchronize update parameters $\theta' = \theta$, $\theta'_v = \theta_v$.
  Get state s_t.
  for $t = 1$, $t \leq t_{max}$ **do**
    Perform a_t according to policy $\pi(a_t \mid s_t; \theta')$.
    Receive reward r_t and new state s_{t+1}.
    $t = t + 1$.
  end for
  Calculate the $R = V(s_t; \theta'_v)$ from the critic network.
  for $t = t_{max}$, $t \geq 1$ **do**
    $R = r_t + \gamma R$.
    Accumulate gradients $d\theta_v$ for critic network by (30).
    Accumulate gradients $d\theta$ for actor network by (31).
    $t = t - 1$.
  end for
  if training at the local training layer **then**
    Asynchronous download and update θ_v and θ from model-chain.
  else
    Send the $d\theta_v$ and $d\theta$ to central consensus layer.
  end if
end while

from surrounding neighboring nodes as input to the model. The local critic network generates the reward value for the last time series location s_t and calculates the cumulative discount function for each moment. Finally, the gradients of the local actor and critic networks are computed separately. During local training, asynchronous training is performed as a local training node running BFA3C, and new model parameters are periodically uploaded to the central control layer.

After receiving the local network models from local training nodes, the central consensus layer obtains topology information from the topology information chain to verify the performance of the local models. The central consensus layer updates the security evaluation value of the local training node based on the model's performance. After integrating the local model, the central consensus layer puts the global model into the block and performs the consensus process. Finally, the local learning nodes get the latest global network model from the model chain and update their local network models. In the next section, we evaluate our proposed BFA3C-based traffic offloading method.

VII. Simulation and Performance Evaluation

A. Simulation Configurations

In this section, we first present the simulated simulation environment and then evaluate our proposed BFA3C traffic offloading algorithm. The SAGIN environment in this paper is shown in Fig. 1, where the space segment consists of LEOs and GEOs, the air segment consists of UAVs, and the ground segment consists of BSs and UEs. The UE is modeled as a source and destination node sending packets and receiving packets, respectively, and other devices are modeled as relay nodes for transmitting packets. There are 100 to 600 source nodes, 30 destination nodes, and 24 BSs on the ground. As a highly mobile device, we refer to the mobility model in [45], where we assign an initial velocity to the UAV, fly in a plane at a certain altitude, and after a period of time arbitrarily change the direction to continue flying with the original velocity. The path loss parameters during UAV communication are set as follows, $(\varphi, \phi_0, \xi, h, k_0)$ is set to (3.04, -3.61, -23.29, 4.14, 20.7) [46]. In addition, we simulate two LEOs moving around the Earth at high speed to provide network access services to the target area periodically. GEO benefit from its wide coverage and provides uninterrupted services to the target area in this environment. In the channel gain of UAV and satellite, F_{rain} is set to 6dB. Table I summarizes the experimental simulation parameters.

TABLE I
Simulation Parameters

Parameters	Values
Number of BS nodes	24
Number of UAV nodes	30
Number of LEO nodes	2
Number of GEO nodes	1
Number of training nodes	8
Simulation Area	$5.76km^2$
Packet generation rate	$N(1e6, \sigma^2)$ Mbps
W_{GEO}	105MHz
W_{LEO}	2.4MHz
W_{UB}	0.5MHz
P_{GEO}	500W
P_{LEO}	200W
P_{UB}	1.6W
G_{US}	10dB
G_{SU}	30dB
BS to BS	10 Mbps
Discount factor	0.9
Max_neighbor_nodes	4
Learning rate	0.001
Exploration probability	1e-8
Block size	1MB
$\boldsymbol{k}$	2.4GHz
C_s	0.034M (CPU cycles)
C_m	4M (CPU cycles)
C_v	4M (CPU cycles)

We first conduct experiments using the Federated-A3C (FA3C) based traffic offloading method without malicious nodes and without considering the blockchain, and use it as a benchmark for comparison in later experiments. Then, to evaluate the performance of our proposed BFA3C-based traffic offloading approach, we compare the BFA3C with the FA3C approach by adding malicious nodes to the SAGIN environment. We mainly evaluate the packet drop rate, latency, and throughput performance.

B. Performance Evaluation

We first evaluate the packet drop rate and the throughput of each method as it grows over time. We set up 340 source nodes, eight local training nodes, and four malicious nodes

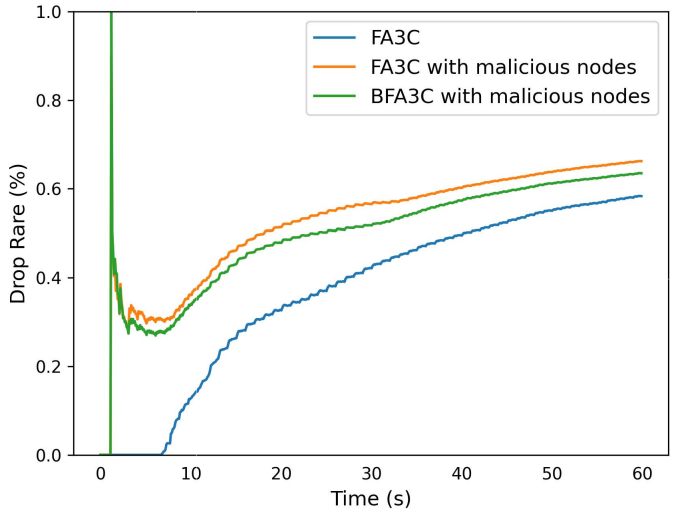


Fig. 5. Packet drop rate over time.

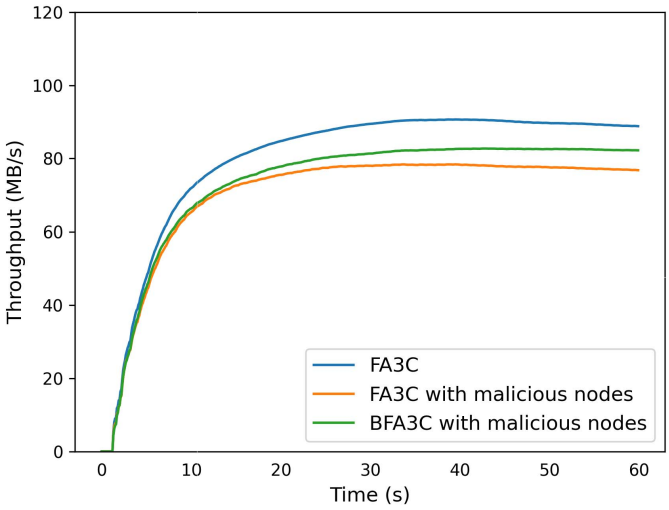


Fig. 6. Network throughput over time.

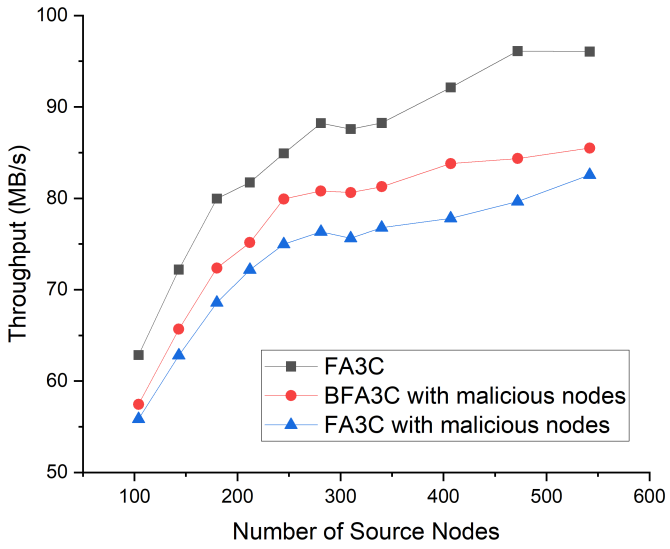


Fig. 7. Throughput per increasing source node count.

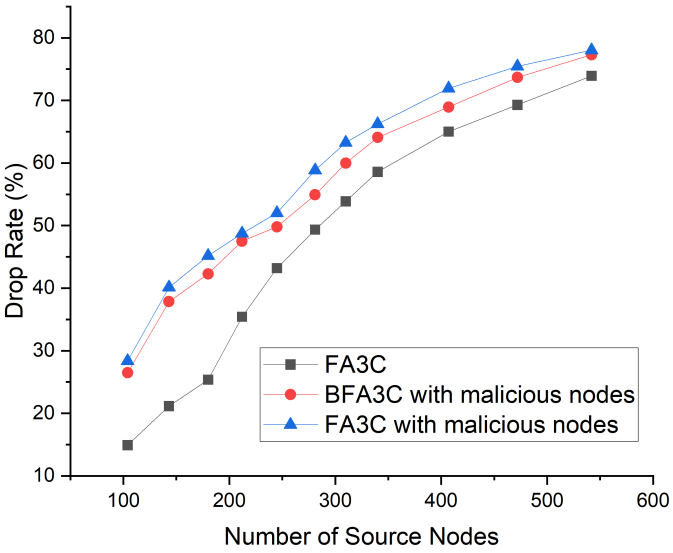


Fig. 8. Drop rate per increasing source node count.

in our environment. Fig. 5 demonstrates the superiority of our proposed BFA3C-based approach in terms of packet drop rate. Since in SAGIN without malicious nodes, individual nodes do not generate malicious behaviors such as dropping packets randomly, the FA3C-based traffic offloading method shows excellent low packet drop rate performance. However, in SAGIN with malicious nodes, the packet drop rate of the FA3C-based approach without blockchain empowerment increases sharply. This is due to the malicious packet drop and arbitrary transmission of packets by malicious nodes that impact the network's overall packet drop rate. In addition, our proposed BFA3C-based traffic offloading method evaluates the security of nodes and gives preference to relay nodes with high security in performing offloading for packet transmission. Therefore, our proposed BFA3C-based approach can effectively reduce the impact of malicious nodes on the network packet drop rate.

The throughput shown in Fig. 6 indicates that the BFA3C-based approach is much closer to the performance of the FA3C-based approach in a malicious node-free environment. There is not much difference in throughput during the initial short period since the data volume is not large enough. As time passes and the number of packets generated by the source node increases, the superiority of the BFA3C-based approach is presented.

Fig. 7, 8, and 9 further demonstrate the superior performance of the BFA3C-based traffic offloading method in the malicious node environment. As the number of source nodes increases and the ground segment is overloaded, we can see from Fig. 7 that the throughput of the BFA3C-based traffic offloading method is consistently higher than that of the FA3C-based traffic offloading method in the malicious node environment.

Fig. 8 shows the variation of the packet drop rate as the number of source nodes increases. As can be seen in Fig. 8,

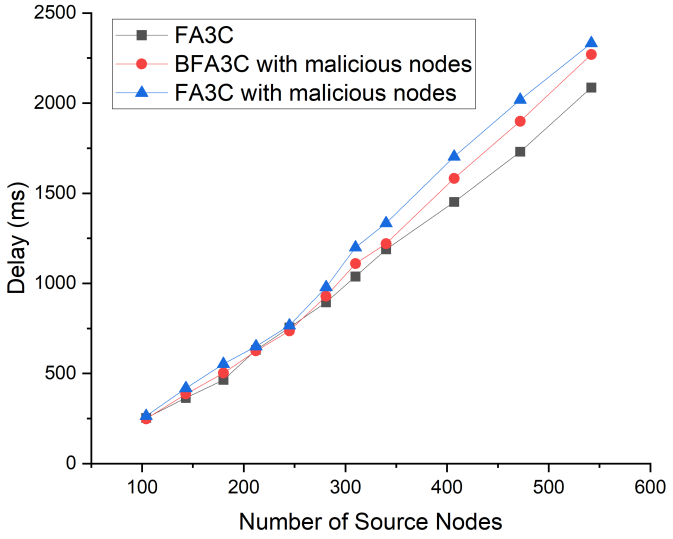


Fig. 9. Packet delay per increasing source node count.

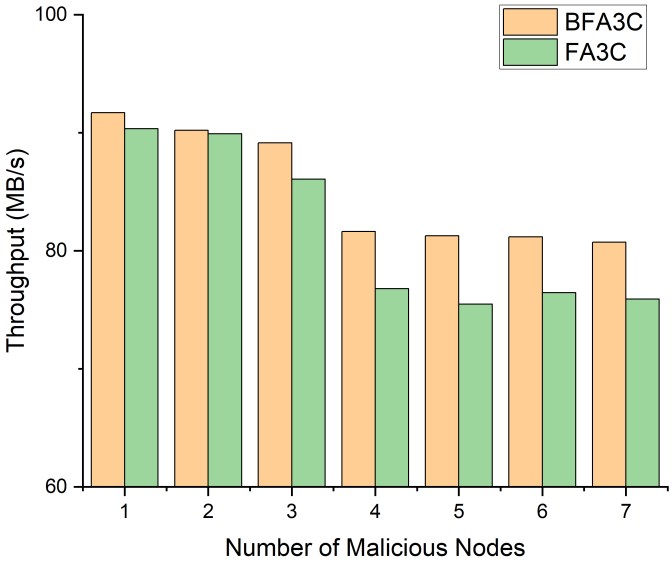


Fig. 11. Throughput per increasing malicious node count.

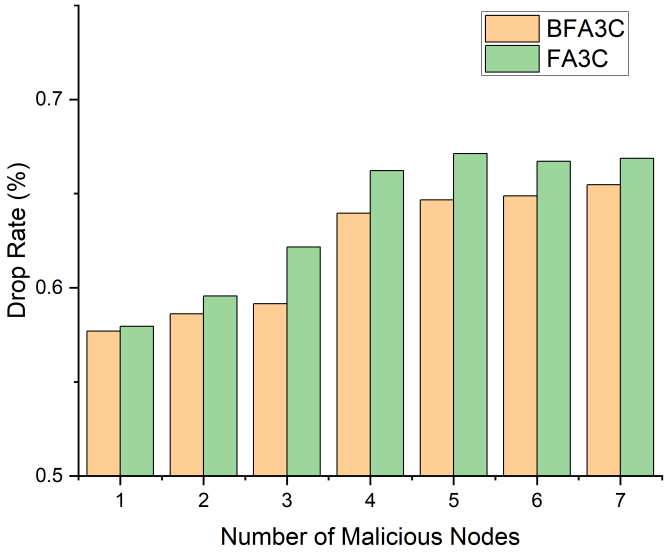


Fig. 10. Drop rate per increasing malicious node count.

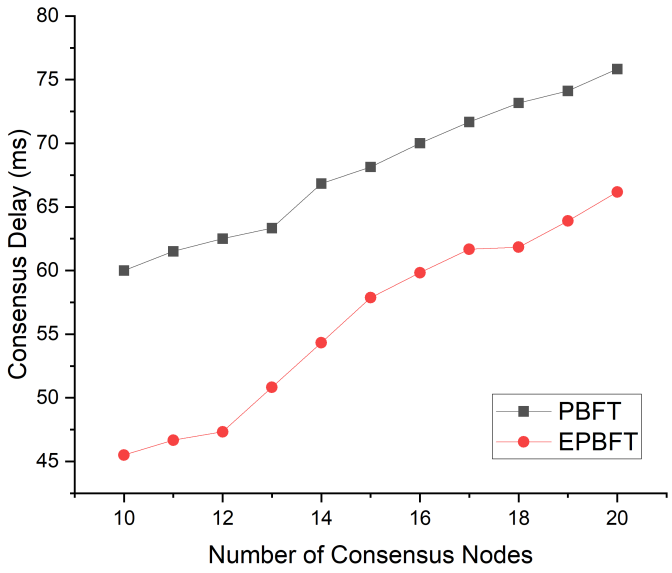


Fig. 12. Consensus delay per increasing consensus node count.

the FA3C-based approach in the malicious node environment has the highest packet drop rate. This is because as more packets are generated, more packets are forwarded through the malicious nodes, resulting in many packets being maliciously dropped. However, the BFA3C-based traffic offloading method effectively avoids the selection of malicious nodes, thus reducing the packet drop rate. As the source node grows to about 550, the packet drop rate also plateaus. This further demonstrates the effectiveness of the BFA3C-based approach in the traffic offloading process.

Fig. 9 shows the effectiveness of the three methods in coping with network delays as the number of source nodes increases leading to an increasing network load. We can see that the BFA3C-based traffic offloading approach is closer to the FA3C-based traffic approach in the malicious node-free environment. In addition, the FA3C-based approach without blockchain empowerment has the highest latency in the malicious node environment. This is because malicious nodes transmit packets to next-hop nodes under non-optimal paths, which leads to an increase in packet transmission latency.

In addition, we evaluate the packet drop rate and throughput of the BFA3C-based traffic offloading method as the number of malicious nodes increases. As shown in Fig. 10 and Fig. 11, the performance of both approaches degrades as the number of malicious nodes increases. Fig. 10 demonstrates the performance of the BFA3C-based approach, which is consistently lower than the FA3C-based approach in terms of packet drop rate. Fig. 11 shows that the BFA3C-based approach is consistently higher than the FA3C-based approach in terms of throughput.

Furthermore, to evaluate the performance of our proposed EPBFT, we compared it with the traditional PBFT algorithm. We set up three malicious nodes in the experimental

environment and evaluated the consensus delay as a performance metric by increasing the number of consensus nodes. As shown in Fig. 12, when the number of consensus nodes is 10, the performance of our proposed EPBFT algorithm is significantly superior to that of the traditional PBFT algorithm. With the increase of consensus nodes, the consensus delay brought by the EPBFT algorithm is significantly lower than the consensus delay brought by the traditional PBFT algorithm.

VIII. Conclusion

Space-Air-Ground Integrated networks have attracted extensive academic and industrial attention as an important component of future intelligent networks. In this paper, we consider traffic offloading works in SAGIN in the existence of malicious node behavior. We demonstrate a novel blockchain-based federated learning architecture for SAGIN with malicious nodes and propose a dual-chain structure for storing topology and model information for the traffic offloading work. To prevent the traffic offloading process from being compromised by malicious nodes, we further propose a node security evaluation mechanism and an enhanced practical byzantine fault tolerance algorithm. We then model the traffic offloading problem as an MDP problem and propose a blockchain-based federated reinforcement learning algorithm. Finally, we evaluate our proposed method and the results show that our proposed method has superior performance in terms of throughput, packet drop rate, and delay.

References

[1] S. Dang, O. Amin, B. Shihada, and M.-S. Alouini, "What should 6G be?" *Nature Electron.*, vol. 3, no. 1, pp. 20–29, Jan. 2020.

[2] L. Bai, R. Han, J. Liu, J. Choi, and W. Zhang, "Relay-aided random access in space-air-ground integrated networks," *IEEE Wireless Commun.*, vol. 27, no. 6, pp. 37–43, Dec. 2020.

[3] J. Liu, Y. Shi, Z. M. Fadlullah, and N. Kato, "Space-air-ground integrated network: A survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2714–2741, May 2018.

[4] N. Kato et al., "Optimizing space-air-ground integrated networks by artificial intelligence," *IEEE Wireless Commun.*, vol. 26, no. 4, pp. 140–147, Aug. 2019.

[5] H. Yang, J. Yang, B. Zhang, and C. Wang, "Visualization analysis of research on unmanned-platform based battlefield situation awareness," in *Proc. IEEE Int. Conf. Artif. Intell. Comput. Appl. (ICAICA)*, Jun. 2021, pp. 334–338.

[6] B. Li et al., "A system of power emergency communication system based BDS and LEO satellite," in *Proc. Comput., Commun. IoT Appl. (ComComAp)*, 2021, pp. 286–291.

[7] Y. Wang, S. Wu, J. Jiao, P. Yang, and Q. Zhang, "On the prediction policy for timely status updates in space-air-ground integrated transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2716–2726, Mar. 2021.

[8] F. Tang, Y. Zhou, and N. Kato, "Deep reinforcement learning for dynamic uplink/downlink resource allocation in high mobility 5G HetNet," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 12, pp. 2773–2782, Dec. 2020.

[9] Z. Lv, A. K. Singh, and J. Li, "Deep learning for security problems in 5G heterogeneous networks," *IEEE Netw.*, vol. 35, no. 2, pp. 67–73, Mar. 2021.

[10] Y. Liu, X. Yuan, Z. Xiong, J. Kang, X. Wang, and D. Niyato, "Federated learning for 6G communications: Challenges, methods, and future directions," *China Commun.*, vol. 17, no. 9, pp. 105–118, Sep. 2020.

[11] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities, and challenges," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 46–51, Jun. 2020.

[12] G. Gui, M. Liu, F. Tang, N. Kato, and F. Adachi, "6G: Opening new horizons for integration of comfort, security, and intelligence," *IEEE Wireless Commun.*, vol. 27, no. 5, pp. 126–132, Oct. 2020.

[13] F. Tang, B. Mao, Y. Kawamoto, and N. Kato, "Survey on machine learning for intelligent end-to-end communication toward 6G: From network access, routing to traffic control and streaming adaption," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1578–1598, Apr. 2021.

[14] L.-Y. Chen, T.-C. Chiu, A.-C. Pang, and L.-C. Cheng, "FedEqual: Defending model poisoning attacks in heterogeneous federated learning," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2021, pp. 1–6.

[15] T.-C. Chiu, W.-C. Lin, A.-C. Pang, and L.-C. Cheng, "Dual-masking framework against two-sided model attacks in federated learning," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2021, pp. 1–6.

[16] S. Zhou, H. Huang, W. Chen, P. Zhou, Z. Zheng, and S. Guo, "PIRATE: A blockchain-based secure framework of distributed machine learning in 5G networks," *IEEE Netw.*, vol. 34, no. 6, pp. 84–91, Nov. 2020.

[17] Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng, and Q. Yan, "A blockchain-based decentralized federated learning framework with committee consensus," *IEEE Netw.*, vol. 35, no. 1, pp. 234–241, Jan. 2021.

[18] T. Hewa, G. Gur, A. Kalla, M. Ylianttila, A. Bracken, and M. Liyanage, "The role of blockchain in 6G: Challenges, opportunities and research directions," in *Proc. 2nd 6G Wireless Summit (6G SUMMIT)*, Mar. 2020, pp. 1–5.

[19] J. Du, C. Jiang, H. Zhang, Y. Ren, and M. Guizani, "Auction design and analysis for SDN-based traffic offloading in hybrid satellite-terrestrial networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2202–2217, Oct. 2018.

[20] J. Li, K. Xue, D. S. L. Wei, J. Liu, and Y. Zhang, "Energy efficiency and traffic offloading optimization in integrated satellite/terrestrial radio access networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 4, pp. 2367–2381, Apr. 2020.

[21] B. Fan, L. Jiang, Y. Chen, Y. Zhang, and Y. Wu, "UAV assisted traffic offloading in air ground integrated networks with mixed user traffic," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 12601–12611, Aug. 2021.

[22] W. C. Ng et al., "Stochastic coded offloading scheme for unmanned aerial vehicle-assisted edge computing," *IEEE Internet Things J.*, early access, Feb. 10, 2022, doi: 10.1109/JIOT.2022.3150472.

[23] B. Mao, F. Tang, Y. Kawamoto, and N. Kato, "Optimizing computation offloading in satellite-UAV-served 6G IoT: A deep learning approach," *IEEE Netw.*, vol. 35, no. 4, pp. 102–108, Jul. 2021.

[24] F. Tang, H. Hofner, N. Kato, K. Kaneko, Y. Yamashita, and M. Hangai, "A deep reinforcement learning-based dynamic traffic offloading in space-air-ground integrated networks (SAGIN)," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 276–289, Jan. 2022.

[25] L. Zhang, W. Abderrahim, and B. Shihada, "Heterogeneous traffic offloading in space-air-ground integrated networks," *IEEE Access*, vol. 9, pp. 165462–165475, 2021.

[26] W. Sun, L. Wang, P. Wang, and Y. Zhang, "Collaborative blockchain for space-air-ground integrated networks," *IEEE Wireless Commun.*, vol. 27, no. 6, pp. 82–89, Dec. 2020.

[27] B. Li, R. Liang, W. Zhou, H. Yin, H. Gao, and K. Cai, "LBS meets blockchain: An efficient method with security preserving trust in SAGIN," *IEEE Internet Things J.*, vol. 9, no. 8, pp. 5932–5942, Apr. 2022.

[28] Z. Guo, J. Zhang, Z. Gao, A. Wang, C. Pan, and X. Li, "Blockchain-based multi-party cooperation and resource-sharing scheme for space-air-ground integrated networks," in *Proc. IEEE 21st Int. Conf. Commun. Technol. (ICCT)*, Oct. 2021, pp. 947–952.

[29] Y. Wang, Z. Su, J. Ni, N. Zhang, and X. Shen, "Blockchain-empowered space-air-ground integrated networks: Opportunities, challenges, and solutions," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 160–209, Dec. 2022.

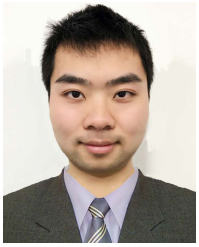
[30] S. Shinde, A. Bozorgchenani, D. Tarchi, and Q. Ni, "On the design of federated learning in latency and energy constrained computation offloading operations in vehicular edge computing systems," *IEEE Trans. Veh. Technol.*, vol. 71, no. 2, pp. 2041–2057, Feb. 2022.

[31] S. Kim, "Cooperative federated learning-based task offloading scheme for tactical edge networks," *IEEE Access*, vol. 9, pp. 145739–145747, 2021.

[32] Y. Zhu, J. Xu, Y. Xie, J. Jiang, X. Yang, and Z. Li, "Dynamic task offloading in power grid Internet of Things: A fast-convergent federated learning approach," in *Proc. IEEE 6th Int. Conf. Comput. Commun. Syst. (ICCCS)*, Apr. 2021, pp. 933–937.

[33] N. Cheng et al., "Space/aerial-assisted computing offloading for IoT applications: A learning-based approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1117–1129, Mar. 2019.

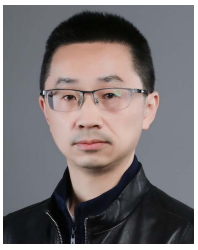
[34] C. Zhou et al., "Delay-aware IoT task scheduling in space-air-ground integrated network," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.
[35] Z. Wang, H. Yu, S. Zhu, and B. Yang, "Curriculum reinforcement learning-based computation offloading approach in space-air-ground integrated network," in *Proc. 13th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2021, pp. 1–6.
[36] Y. Cao, S.-Y. Lien, Y.-C. Liang, and K.-C. Chen, "Federated deep reinforcement learning for user access control in open radio access networks," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2021, pp. 1–6.
[37] M. Zhang, Y. Jiang, F.-C. Zheng, M. Bennis, and X. You, "Cooperative edge caching via federated deep reinforcement learning in fog-RANs," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, Jun. 2021, pp. 1–6.
[38] R. Ali, Y. B. Zikria, S. Garg, A. K. Bashir, M. S. Obaidat, and H. S. Kim, "A federated reinforcement learning framework for incumbent technologies in beyond 5G networks," *IEEE Netw.*, vol. 35, no. 4, pp. 152–159, Jul. 2021.
[39] A. Al-Hourani and K. Gomez, "Modeling cellular-to-UAV path-loss for suburban environments," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 82–85, Feb. 2017.
[40] S. A. Kanellopoulos, C. I. Kourogiorgas, A. D. Panagopoulos, S. N. Livieratos, and G. E. Chatzarakis, "Channel model for satellite communication links above 10 GHz based on Weibull distribution," *IEEE Commun. Lett.*, vol. 18, no. 4, pp. 568–571, Apr. 2014.
[41] H. Liao et al., "Blockchain and semi-distributed learning-based secure and low-latency computation offloading in space-air-ground-integrated power IoT," *IEEE J. Sel. Topics Signal Process.*, p. 1, 2021.
[42] M. Castro and B. Liskov, "Practical byzantine fault tolerance," in *Proc. OSDI*, vol. 99, 1999, pp. 173–186.
[43] A. Clement, E. L. Wong, L. Alvisi, M. Dahlin, and M. Marchetti, "Making byzantine fault tolerant systems tolerate byzantine faults," in *Proc. NSDI*, vol. 9, 2009, pp. 153–168.
[44] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.
[45] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Commun. Mobile Comput.*, vol. 2, no. 5, pp. 483–502, Aug. 2002.
[46] W. Shi et al., "Multi-drone 3-D trajectory planning and scheduling in drone-assisted radio access networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8145–8158, Aug. 2019.



Fengxiao Tang (Member, IEEE) received the B.E. degree in measurement and control technology and instrument from the Wuhan University of Technology, Wuhan, China, in 2012, the M.S. degree in software engineering from Central South University, Changsha, China, in 2015, and the Ph.D. degree from the Graduate School of Information Science, Tohoku University, Japan. He was an Assistant Professor from 2019 to 2020 and an Associate Professor from 2020 to 2021 at the Graduate School of Information Sciences (GSIS), Tohoku University. He is currently a Full Professor with the School of Computer Science and Engineering, Central South University. His research interests include unmanned aerial vehicles systems, the IoT security, game theory optimization, network traffic control, and machine learning algorithm. He was a recipient of the Prestigious Dean's and President's Awards from Tohoku University in 2019 and several Best Paper Awards from conferences, including IC-NIDC 2018 and GLOBECOM 2017/2018. He was also a recipient of the Prestigious Funai Research Award in 2020, the IEEE ComSoc Asia–Pacific (AP) Outstanding Paper Award in 2020, and the IEEE ComSoc AP Outstanding Young Researcher Award in 2021.

Cong Wen (Student Member, IEEE) received the B.E. degree in software engineering from the East China University of Technology, Nanchang, China, in 2019. He is currently pursuing the M.S. degree in software engineering with the School of Computer Science and Engineering, Central South University, Changsha, China. His research interests include space-air-ground integration networks, vehicular networks, and machine learning in wireless networks.

Linfeng Luo (Student Member, IEEE) received the B.S. degree in computer science and technology from the Huazhong University of Science and Technology, Wuhan, China, in 2019. He is currently pursuing the M.S. degree in software engineering with the School of Computer Science and Engineering, Central South University, Changsha, China. His research interests include network traffic control and machine learning algorithm.



Ming Zhao (Member, IEEE) received the M.Sc. and Ph.D. degrees in computer science from Central South University, Changsha, China, in 2003 and 2007, respectively. He is currently a Professor with the School of Computer Science and Engineering, Central South University. His main research interests include wireless networks. He is also a member of the China Computer Federation.

Nei Kato (Fellow, IEEE) is currently a Full Professor and the Dean of the Graduate School of Information Sciences, Tohoku University. He has published more than 500 papers in prestigious peer-reviewed journals and conferences. His research interests include computer networking, wireless mobile communications, satellite communications, ad hoc and sensor and mesh networks, smart grid, AI, the IoT, big data, and pattern recognition. He is a fellow of the Engineering Academy of Japan and IEICE. He was the Vice President (Member and Global Activities) of the IEEE Communications Society from 2018 to 2019 and the Editor-in-Chief of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY from 2017 to 2020 and the IEEE NETWORK from 2015 to 2017.