Received: 12 May 2021 | Revised: 23 September 2021 | Accepted: 28 September 2021

DOI: 10.1002/cpe.6700

SPECIAL ISSUE PAPER

WILEY

SBHA: An undetectable black hole attack on UANET in the sky

Runqun Xiong[1] | **Lan Xiong**[2] | **Feng Shan**[1] | **Junzhou Luo**[1]

[1]School of Computer Science and Engineering, Southeast University, Nanjing, China

[2]School of Cyber Science and Engineering, Southeast University, Nanjing, China

Correspondence
Runqun Xiong, School of Computer Science and Engineering, Southeast University, Nanjing, China.
Email: rxiong@seu.edu.cn
Lan Xiong, School of Cyber Science and Engineering, Southeast University, Nanjing, China.
Email: lxiong@seu.edu.cn

Funding information
National Natural Science Foundation of China, Grant/Award Numbers: 62172091, 61602112, 61632008; Jiangsu Provincial Key Laboratory of Network and Information Security, Grant/Award Number: BM2003201; Key Laboratory of Computer Network and Information Integration of the Ministry of Education of China, Grant/Award Number: 93K-9; International S&T Cooperation Program of China, Grant/Award Number: 2015DFA10490; Collaborative Innovation Center of Novel Software Technology and Industrialization and the Collaborative Innovation Center of Wireless Communications Technology

Summary

With their high flexibility and versatility, unmanned aerial vehicles (UAVs) have maneuvered their way into many applications. Thanks to their ability to plan and coordinate, multiple UAVs complete tasks more effectively, which boosts their popularity in battlefield surveys, formation performances, and targeted searches. However, the risk of security threats also rises alongside their popularity. The UAV ad hoc network (UANET) has endeavored to contend with such risks through the optimized link state routing (OLSR) protocol. To test the security and strength of this effort, we present a sky black hole attack (SBHA) algorithm for OLSR, which is undetectable, based on the UANET's multi-hop routing and the OLSR's known topology. This algorithm obtains the network's maximum profits by approaching and then replacing the calculated topology center and traffic center in UANET. Because of the ever-changing topology, SBHA aims at UANET's single central node that cannot be detected in advance. This attack is difficult to detect by UANET and therefore difficult to defend. The simulation results show that SBHA can cause greater damage to UANET compared to a traditional black hole attack, and ordinary defense algorithms cannot reduce the negative impact of SBHA on UANET. In addition, SBHA also gains UANET control, and leads to drastic changes in UAVs' movement trajectory, which has more intuitive effects.

KEYWORDS

black hole attack, NS-3, OLSR, UAV ad hoc network

1 | INTRODUCTION

In recent years, unmanned aerial vehicles (UAVs) have gradually integrated into all quarters of society and become indispensable civilian and military equipment. With this technology's development, the tasks that UAVs need to complete are becoming increasingly complex. Organizing multiple UAVs to complete the mission collaboratively is an important task of UAV applications. The key to realizing multiple UAVs' collaboration technology lies in the communication between UAVs. Therefore, the basis of this technology is the communication relayed via the UAV ad hoc network (UANET).

UANET is a model proposed for the characteristics of the communication topology of multiple UAVs' collaboration. In UANET, the communications between UAVs do not completely rely on basic communication facilities such as ground control stations or satellites. Instead, UAVs are used as network nodes. Every node can exchange data such as a perception situation, health status, and intelligence collection to each other, and automatically connect to establish a wireless mobile network. Each node in the network functions as a transceiver and a router. UAV is not only a node in the network, but also acts as a relay node in the ad hoc network with a routing function, forwarding data to nodes that are farther away via multi-hop transmission.

The UAV network, constructed in the form of an ad hoc network, needs the routing protocol's support. The optimized link state routing (OLSR) protocol[1] is a table-driven routing protocol widely used in UANET.[2] In UANET, based on OLSR, the node maintains the routing table, and thus

network latency is low. Additionally, the node regularly updates the routing table to reduce the impact of topology fluctuations on network connectivity. Compared to the classic LSR protocol, OLSR adds its main idea—a multipoint relay (MPR) mechanism—for optimization. When LSR uses a flood-propagation technology, the node that receives any message must transmit and forward the data packet to all its neighbors to ensure that every node in the network understands the network topology. In OLSR, every node sending topology control (TC) messages and packets must be forwarded, depending on the MPR set selected, which reduces network flooding and also makes the MPR nodes' status in the network more important. So, the local node can grasp the remote topology and discriminate the maximum traffic node to achieve the best profits in OLSR.[3] It then becomes difficult to defend against an attack in UANET during a period of high mobility. The UANET based on the table-driven routing protocol has the advantages of low data packet delay, and the error of the real-time accuracy of topology information can be ignored. Therefore, although the routing protocol is more suitable for UANETs, Corresponding improvements are needed, and it is still feasible to choose this protocol as the routing protocol for the UANET.[4]

Keeping this in mind, here we propose a sky black hole attack (SBHA) algorithm for UANET based on OLSR, which is undetectable for the existing defense measures, depending on the multi-hop routing of UANET and the known topology of OLSR. We also optimize the attack algorithm according to the UAV characteristics, and finally achieve the goal of gaining network control. In SBHA, the malicious node calculates the topology and traffic central nodes in UANET through the routing table, moves to the position where the maximum traffic can pass through the network, and uses the real MPR identity as a neighbor to absorb, modify, and discard packets. In this way, the malicious node can change the UAVs' formation and trajectory (which causes greater damage to the network) and finally gain network control.

The following are the main contributions of this article:

1. SBHA uses OLSR's MPR mechanism to attack without sending forged messages to deceive the network. Instead, it absorbs packets by occupying the network's central node position. The black hole attack prevention and defense measures cannot deter SBHA's impact on UANET.
2. SBHA uses UANET's multi-hop routing and the known topology of OLSR to calculate the network's topological center and traffic center with the largest number of packets currently passing through, and set it as the attack's focus. Compared to traditional black hole attacks that do not perform calculations and attacks on the central node, SBHA has a wider attack range, so it can cause greater damage to UANET and profits from the attack more effectively.
3. SBHA has researched the UAVs' movement information, making this algorithm more suitable for UANET with high mobility. The experimental results indicate that SBHA can change the movement trajectory and working status of UAVs at the same time. In addition, SBHA also obtains UANET control, allowing it to view the effects of attacks more intuitively.
4. The optimized realization of the black hole attack by this algorithm has produced obvious and effective attack effects on UANET, which are applicable to and scalable for other mobile ad hoc networks.

The remainder of this article is organized as follows. Section 2 discusses work related to the attack and defense for a mobile ad hoc network (MANET). Section 3 presents our SBHA algorithm. Section 4 discusses the experimental results based on NS-3. Finally, Section 5 summarizes our work and indicates future research directions.

2 | RELATED WORK

Secure routing plays a pivotal role in reliable network performance. Most conventional secure-routing protocols[5-7] assume that there is a centralized or distributed third party in the network to implement multiple encryption algorithms and ensure routing security just as Zhao et al.[8] apply lightweight two-factor authentication to ensure the security of the Internet of Things. However, these assumptions are invalid for MANET. Because of the network's lack of fixed infrastructure, open transmission media, and dynamic topology, MANET has the most unfavorable networking environment and lacks dependence on prerequisites. In addition, on resource-constrained mobile nodes, cryptographic operations such as computing digital signatures and verification will consume too much of the mobile node's resources. The most important problem in these routing protocols is that they cannot recognize certain passive attacks, such as black hole attacks.

In Deng et al.,[9] the authors conducted thorough research on black hole attacks for the ad hoc on-demand distance vector (AODV) protocol.[10] In AODV, the malicious node replies to every routing request it receives, instead of only replying to the route whose destination node is reachable. Then, the malicious node discards all the data packets passing through it and does not help to forward according to the protocol's requirements. This creates a black hole in the network, which is easily ported into OLSR by declaring a forged Hello message about false neighbor information to increase its chances of being selected as an MPRAODV.[11]

Another form of black hole attack is the tc-hello-black-hole.[12] By forging Hello and TC messages at the same time, an attacker can ensure that packets will flow through it. However, there is no guarantee that the attack will proceed effectively, because the packet may actually choose a different route.

Raffo et al.[13] proposed a mechanism to improve OLSR security for the aforementioned attacks. In this solution, every node signs its Hello and TC messages. Then these signatures are used by others to prove their Hello and TC messages. This scheme can prevent the malicious devices from claiming virtual connections with known nodes and spoofing links.

However, when the attacker is a natural candidate for transmitting packets or the attacker is nominated as the MPR of other nodes, the attack will fail, because the user can still discard packets passing through it. In addition, this scheme's mechanism also consumes a lot in signing messages. As the network's scale increases, the accumulation factor also will increase.

Schweitzer et al.[14] proposed a trust-degree mechanism algorithm to minimize gray hole denial-of-service (DoS) attacks. By checking whether the Hello message packet's content (sent by each node in the network) contradicts the known topology message, the trusted node is divided from the suspicion. We use five different threat models to evaluate the technology, which gives us a better understanding of the attack surface and preventive measures. The final results show that this algorithm greatly reduces the gray hole attacks' efficiency.

Similarly, Cai et al.[15] proposed an Evolutionary Self-Cooperative Trust (ESCT) scheme to prevent black hole attacks in MANET, which mimics the human cognitive process and relies on trust-level information to prevent various routing interruption attacks. In this scheme, mobile nodes will exchange trust information and analyze the received trust information based on their own cognitive judgment. In the end, each node will dynamically develop its cognitive capabilities to exclude malicious nodes.

Algorithms of this kind are highly effective for traditional black hole attacks' defense, because none of the black hole attacks mentioned can identify the network center. Instead, they create a virtual one. A physical network center—assuming it is discovered by an attacker—remains unprotected.[16]

Schweitzer et al.[17] proposed a method to find the nodes that must be passed through to forward some packets in OLSR-based MANET if the nodes exist, with a linear cost. After locating a node that can get the most revenue, the prevention methods can increase the nodes' protection measures to prevent attackers from attacking and hijacking the nodes with greatest profits. However, this method can only find bottleneck nodes in the local path, and does not find the node with the greatest profits for the entire network. Additionally, this method has limitations for networks with ever-changing topology.

For the SBHA algorithm proposed in this article—in which malicious nodes did not try to send false Hello message to deceive the network; instead, they calculated an approach for the largest data traffic in real time—the aforementioned defense scheme is not effective.

Existing studies have performed substantial work on black hole attacks and gray hole attacks deployed on MANET, but they have not combined specific scenarios such as UANET with attack algorithms for vulnerability research.[18] And the most extensive defense scheme currently is to judge whether the node is trustworthy based on the degree of trust. This kind of scheme has no practical defense function for SBHA.

3 | SKY BLACK HOLE ATTACK

In a classic black hole attack, the attacker should claim that all nodes in the topology table are his neighbors, to cheat an entire network's nodes by broadcasting a forged Hello message. While SBHA does not need to send forged Hello messages, it does capture the information of all nodes directly by replacing the central node. This section divides the main concept of SBHA into two parts: capturing packets and tampering with packets. Capturing packets' module can be divided into calculating, approaching, and replacing the central node. Figure 1 shows the SBHA algorithm's framework.

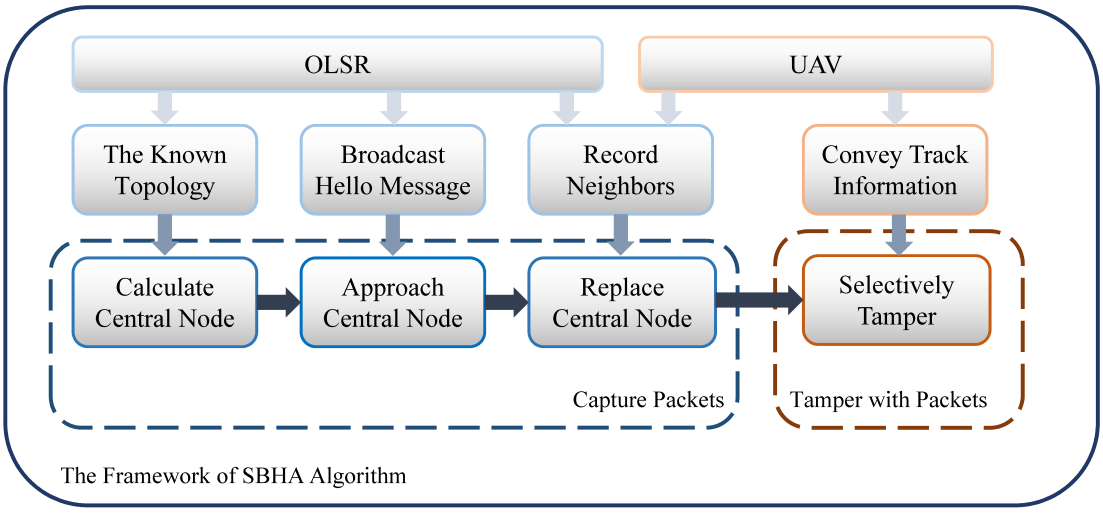

FIGURE 1 Sky black hole attack (SBHA) algorithm framework. OLSR stands for optimized link state routing protocol and UAV stands for unmanned area vehicle

3.1 | Capturing the packets module

If an attacker wants to capture packets as much as possible and does not adopt easy-to-detect methods such as broadcasting a forged Hello message, then first malicious node should be the center of network and become a "hub" that can help all nodes truly transmit packets. However, the malicious node only can obtain the routing table through OLSR, and cannot know the specific traffic center. Therefore, we use it to calculate the current network topology's central and traffic center nodes that pass the most packets currently through the routing table and multi-hop routing information. Then, according to the calculation results and type of data packet flow, the attacker approaches the node that can obtain the most profits. Finally, the UAV feature is used to destroy the node's connectivity, occupy its position as the network's main hub, and become the UANET's new central node.

Using this scheme, most existing detection methods can be avoided, so that the malicious node can obtain and exploit the real MPR identity, thereby further realizing the black hole's function, changing the UAV's movement trajectory, and seizing network control. Figure 2 shows the flowchart of the capturing packets' module.

3.1.1 | Calculating the topology and traffic centers

The topology center is the node at the network's center, which theoretically can intercept most packets. The topology center calculated by each node at the same time should be the same. According to the network topology information, the network topology center can be calculated by taking the sum of the distances from each node in the network to the node as a measurement standard.

The resources owned by the malicious node include the topology and routing tables. Because of the topology table's real-time transformation, the topology may be duplicated or missing. Thus, the breadth-first search (BFS) algorithm and judgment of special circumstances are used, to ensure consistency among the network's topology center results calculated by each node. Algorithm 1 is used to find the sum of distances from one node to other nodes.

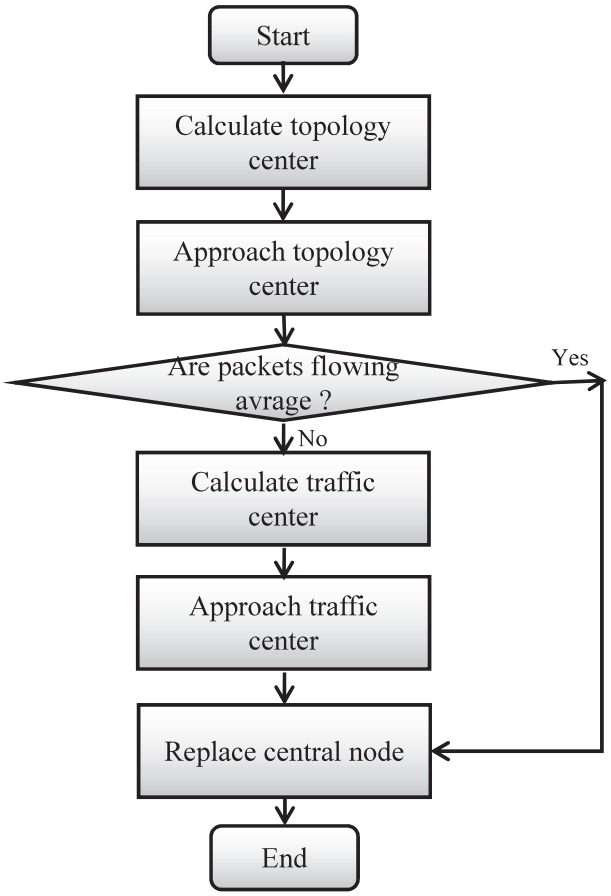

FIGURE 2 Flowchart of the capturing packets module

Algorithm 1. Sum-of-distances

```
Input: Ipv4Address of a node (ip)
Output: sum of distances from this node to others (sum)
  function SUM-OF-DISTANCES(ip)
    if ip == mainAddress then
      return OwnSum
    end if
    if ip is in topology table then
      BFS(ip, sum, topology)
    else
      if ip is in neighbor table then
        sum ← sum + own_sum + routing.size() − neighbor.size()
      else
        sum ← INT_MAX
      end if
    end if
    return sum
  end function
```

The traffic center is the node that can actually intercept the most packets combined with an application layer. The calculation results of the center node will change depending on the time and location. By using multi-hop routing information, the node that passes the most packets is calculated from all the nodes that each packet passes through—from the source node to the destination node.

To calculate the traffic center, when a malicious node sends or receives a packet that needs to be processed and forwarded, it checks the packet's source and destination nodes, and finds and records the path the packet passes. Similarly, because of incomplete information in the topology table, all the links in the network will not be recorded completely. A malicious node cannot calculate the traffic center via topology and routing tables directly, but it needs to use the depth-first search (DFS) algorithm and consider special circumstances. Algorithm 2 is used to get the packet passes' nodes, and nodes that pass through them can be figured out by Algorithm 3.

3.1.2 | Calculating the topology and traffic centers

In UANET, the protocol will spread its own GPS information through Hello messages. Thus, the attacker can approach a certain node via the protocol. Through the NS-3 simulation experiment, it is concluded that the calculated traffic center has certain limitations at the edge position. When the direction is near the topology center, the prediction is more accurate. For an instance with stable packets flowing, the topology center can intercept more packets than the traffic center calculated by most nodes.

Algorithm 2. Nodes-of-packet-passes

```
Input: a packet from the route (packet)
Output: the nodes of the packet passes (nodes)
  function NODES-OF-PACKET-PASSES(packet)
    source ← packet.getSource()
    dest ← packet.getDestination()
    nodes ← nodes + {mainAddress}
    if source! = mainAddress then
      NODES-OF-ADDRESS(source, nodes)
    end if
    if dest! = mainAddress then
      NODES-OF-ADDRESS(dest, nodes)
    end if
    return nodes
  end function
```

Algorithm 3. Nodes-of-address

Input: an address (addr)
Output: the nodes that pass through (nodes)
function NODES-OF-ADDRESS(*addr*)
nodes ← *nodes* + {*addr*}
if *addr* is in neighbor table **then**
return *nodes*
end if
if *addr* is in tow-hop neighbor table **then**
nodes ← *nodes* + {*neighborAddr*}
else
DFS(*addr*, *nodes*, *topology*)
end if
return *nodes*
end function

By comparing the topology center and traffic center calculated with different applications intercepting packets, we divide the nodes close to the center into three situations:

1. For applications where the packets' flow is average and stable, it can be assumed that the topology center is the traffic center, and more packets can be intercepted by directly attacking the topology center.
2. For applications with irregular packets' flow, the attacker can approach the topology center first, and then calculate the packets' flow near the topology center to find the most suitable location to intercept the data packet.
3. For high-mobility applications, it is necessary to constantly adjust the computing topology center and traffic center nodes to intercept more packets.

3.1.3 | Replacing the central node

UAVs are flexible, but they also have limited energy consumption and storage resources.[19] The OLSR message-flooding mechanism causes the nodes in UANET to occupy some storage space. Therefore, we consider broadcasting a large number of Hello message in different IPs through a single hop to occupy the resources of the central node and its neighbors as Huo et al.[20] working for social Internet of Things. In this way, the Hello messages, TC messages, and packets sent by the central node's real neighbors are all overwhelmed by the false Hello messages from the malicious node, thereby destroying the two-way link between the central node and its neighbors. When a malicious node becomes the MPR of its neighbor node, it successfully occupies the central node position.

After replacing the traffic center node, packets can be captured at the largest traffic's location on the entire network. This attack method uses the UAVs' characteristics and the message-flooding mechanism of OLSR, which is extremely difficult to defend in an ad hoc network with average functions of all nodes. In addition, in an NS-3 simulator, it takes an average of 23 s to replace a node, and takes 40–60 s to build a complete routing table. This method is quick and effective.

3.2 | Tampering with the packets module

In SBHA, after the malicious node becomes the new central node in the network, to make UANET's attack effect more obvious, the attacker tampers wit and drops packets selectively according to the analysis of the movement trajectory of each drone instead of dropping all the packets. Figure 3 shows the flowchart for tampering with the packets module.

3.2.1 | UAV motion trajectory analysis

Multiple UAVs' movement can be divided into two states: one where the UAVs maintain the current formation to fly at the same speed, and another that changes the current formation to a new formation. For these two states, the ground control station will send commands to its directly connected UAVs, and these UAVs will load the command into a packet and broadcast it to all the network nodes.

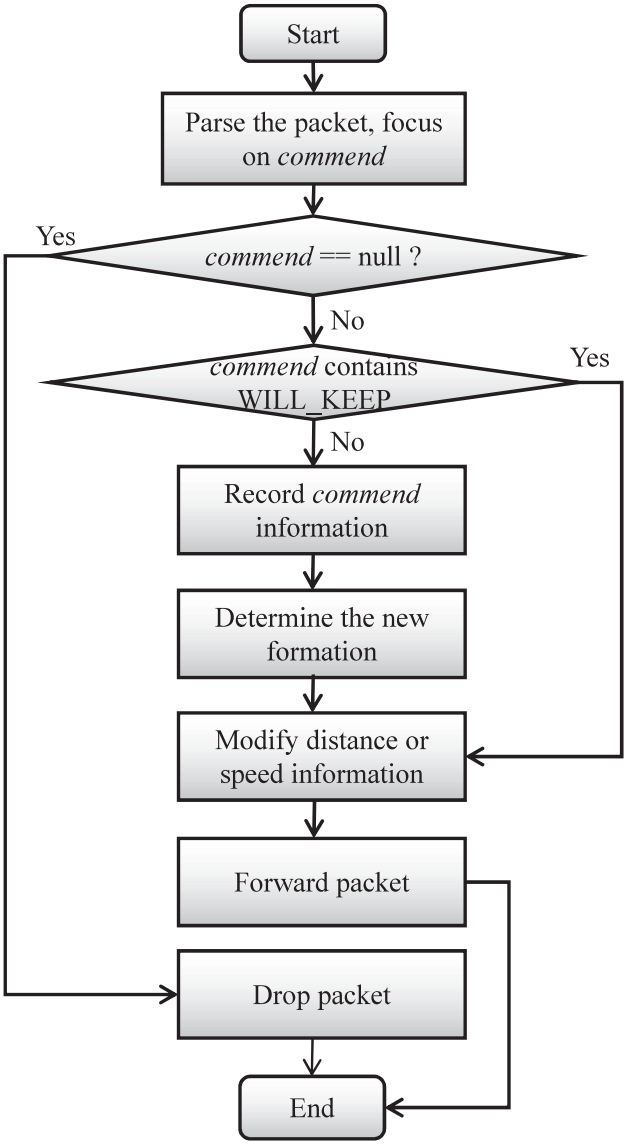

FIGURE 3 Flowchart of tampering with the packets module

When multiple UAVs need to maintain the formation, the first line of "command" is WILL_KEEP, which means that the formation will not change, and the second line indicates the entire formation's direction and speed. Figure 4 shows the maintaining formation state diagram. The UAVs connected to the ground station directly loads the command into the packet and broadcasts it, and the nodes receiving the packet maintain the speed in the corresponding direction according to the command. They don't change speed until receiving the next command.

When multiple UAVs change their formation, the first command line is WILL_CHANGE, which means the formation is about to change. The following lines are the directions and distances calculated by the ground station that each node needs to follow. Each line has an IP address representing a UAV node, with the direction and distance the node needs to move. Figure 5 shows the state diagram of the transforming formation. When UAVs connected to the ground station receives the command, the station checks its routing table according to the IP address. If the node is reachable, it writes WILL_CHANGE along with the distance direction into the packet's commend field and sends it to the corresponding node. The node receiving the packet will move according to the direction in the command until it reaches the destination position. Then it will stop moving and reply to the source node, indicating that it has reached its destination. After receiving the next command, it will change its trajectory.

3.2.2 | Selective packet tampering

After the malicious node intercepts the packet in UANET, it parses the packet and focuses on the commend field. When the commend field is empty—which means that the packet does not carry information about the UAVs' motion trajectory—the attacker will drop it.

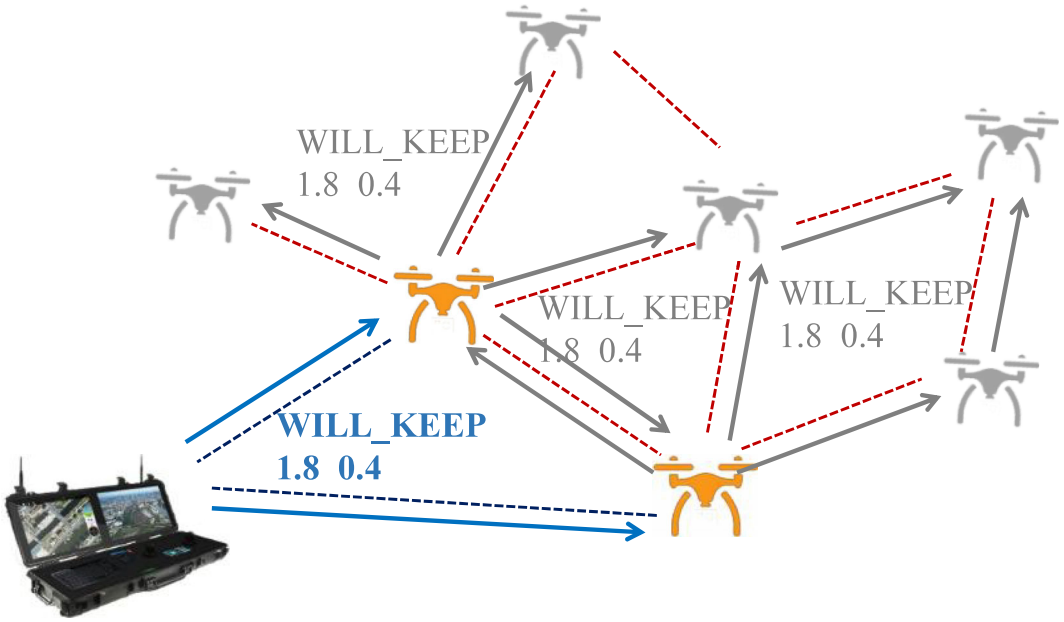

FIGURE 4 Maintaining formation state diagram

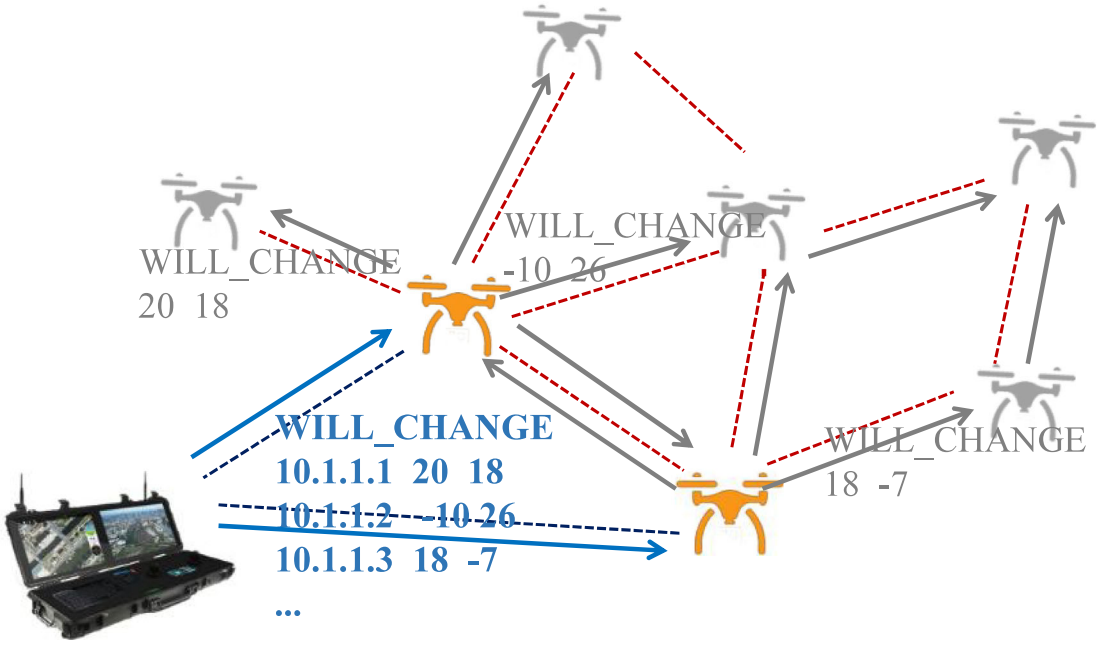

FIGURE 5 Transforming formation state diagram

When the commend field is not empty, if the first command line is WILL_KEEP, it means that the network needs to maintain the formation movement. The malicious node can tamper with the direction and speed of the second command line, or change the first line to WILL_CHANGE. So, the attacker can change UANET's formation.

If the first command line is WILL_CHANGE; it means that the network needs to modify the formation. The malicious node can calculate through the routing table to tamper with the direction and distance, so that the network is modified to an unexpected formation. The attacker can also change the first line to WILL_KEEP, letting the ground station fail to modify the formation.

In SBHA, malicious nodes selectively tamper with or drop packets by analyzing each UAV's movement trajectory to achieve the goal of obtaining network control finally.

4 | EXPERIMENTAL SIMULATION

Because the network topology is infinite, the algorithm must be verified through simulation.[21] In this section, the operation of various simulation experiments is described to prove SBHA's feasibility under a detection method and the ultimate effect of obtaining network control rights for UANET.

In this article, we used NS-3.29[22] to conduct simulation experiments on Ubuntu. Using C++ programming, the network simulator's node information was set to simulate ordinary and malicious UAV nodes. The routing protocol used the built-in OLSR module, and corresponding modules in the third section were added into it. The main difference between a malicious node and a normal node was the m-malicious parameter, and the malicious node ran after the normal node when the simulator is running. In the simulation, the UAVs' speed was 1.5–2 m/s, and transmission range was about 250 m. We also use 802.11 standards to establish network channels in NS-3, with the simulating UAV nodes installed by energy consumption and motion modules.

4.1 | Comparative experiment of capturing packets

Figure 6 shows the UANET's architecture for the black hole attack experiment. The UAV nodes ran the built-in OLSR module in NS-3 at the network layer, simulated the energy consumption process of each UAV through the energy module, simulated UAVs flying in a certain pattern by setting the mobility module information, and forwarded packages through a server-client module. In the "capturing packets" experiment, malicious nodes intercepted the packets sent by ordinary nodes and dropped the packets directly. This section will calculate the packet loss rate and network throughput for the following five situations, respectively, to show SBHA's advantages of comparing it with the classic black hole attack algorithm in the packet-capturing stage versus SBHA's feasibility under a detection method.

NON_BLACKHOLE: Here, there is no black hole attack in the network. The malicious node does the same operation as the normal node. Each node in the network randomly sends packets to other nodes to check the packet loss rate, network throughput, packets delay and network jitter under normal conditions.

BLACKHOLE: In this instance, the malicious node performs a classic black hole attack by adding a hop of unreachable nodes to the Hello messages' neighbor information, capturing packets in the network as much as possible, and then dropping it. Check the packet loss rate, throughput, delay and jitter of the network with a classic black hole attack.

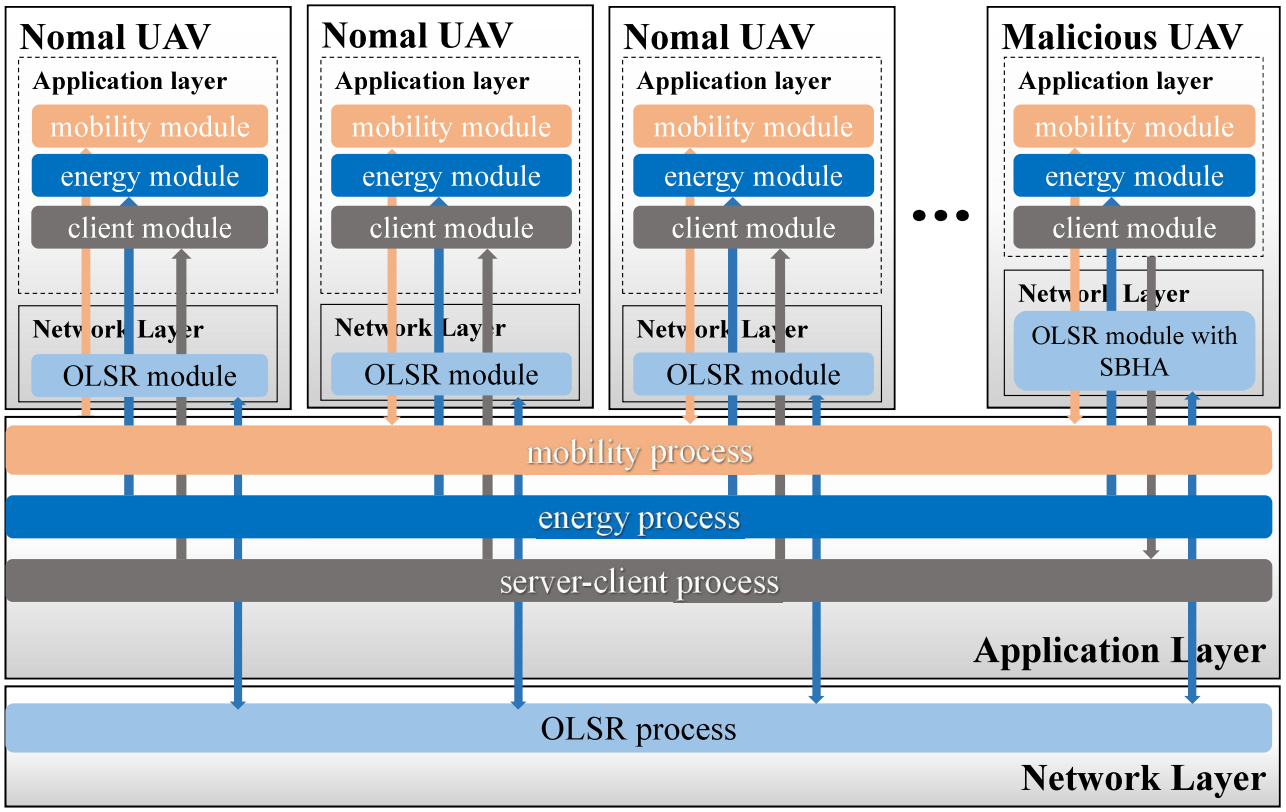

FIGURE 6 Black hole attack experiment's architecture

DETECT_BLACKHOLE: In this case, malicious nodes still perform classic black hole attacks. Simultaneously, common nodes are added with a trustworthiness detection module in OLSR, which checks whether the source node has added one-hop unreachable nodes to a neighboring table for each Hello message received. If a node is suspect, the possibility of the node being selected as the MPR is reduced. Check the packet loss rate, throughput, packets delay and network jitter with a classic black hole attack and prevention method.

SBHA: Here, the malicious node executes an SBHA algorithm proposed in this article, captures and drops packets by calculating and replacing the central node, and checks the packet loss rate, throughput, packets delay and network jitter with SBHA.

DETECT_SBHA: In this scenario, the malicious node still executes the SBHA algorithm proposed in this article. Simultaneously, the common node is added with a trustworthiness detection module in OLSR. When the prevention method has been used, check the packet loss rate, throughput, packets delay and network jitter after using the SBHA algorithm.

According to the five aforementioned situations, Figures 7–10 show the experimental results simulated on the NS-3 simulator.

4.2 | Comparative experiment of tampering with packets

In the SBHA algorithm's tampered packets module, the malicious node classifies the captured packets according to the commend field in packets representing the UAV's movement information. As mentioned previously, there are three types of packets: the first type contains no UAV motion

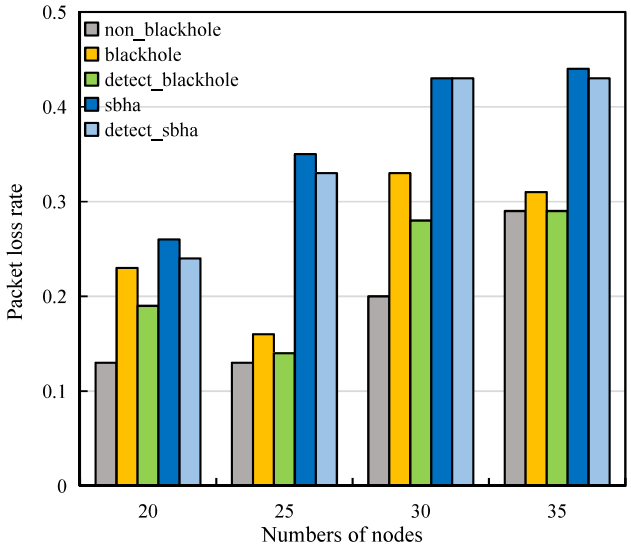

FIGURE 7 Network packet loss rate under different conditions

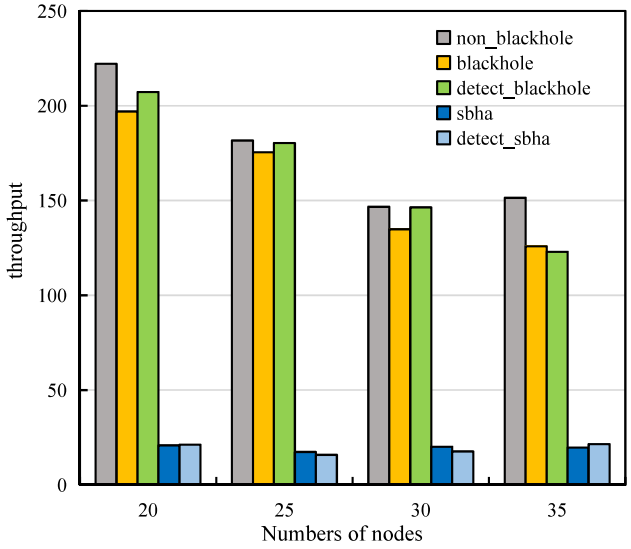

FIGURE 8 Throughput under different conditions

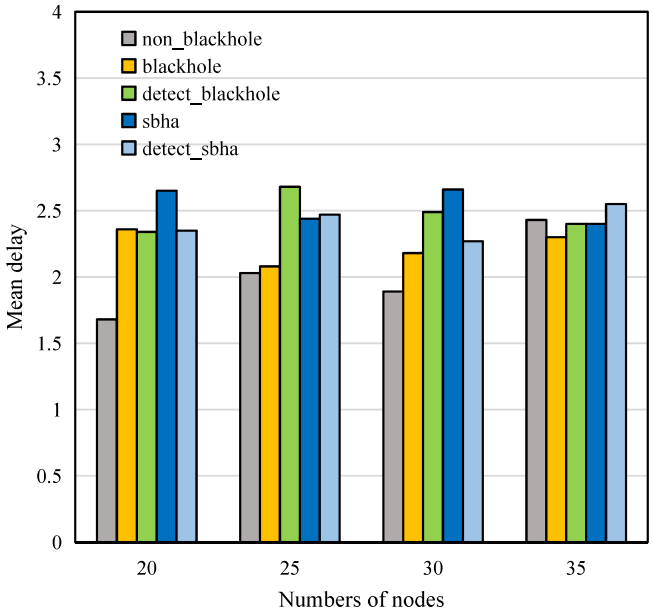

FIGURE 9 Delay under different conditions

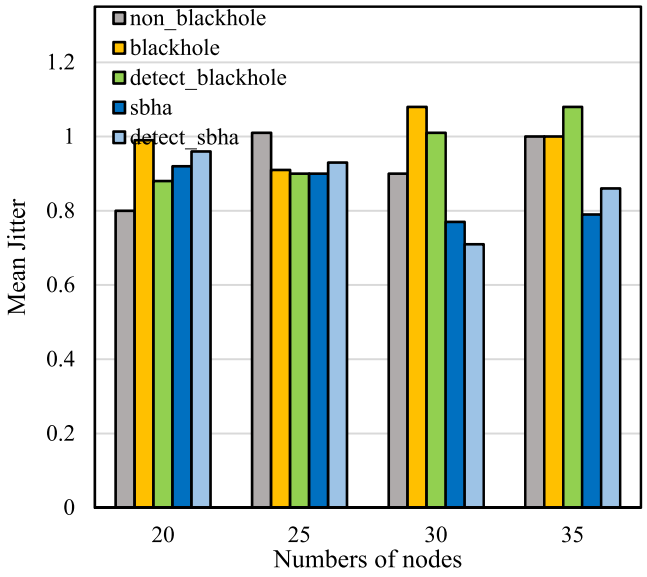

FIGURE 10 Jitter under different conditions

information; the second type indicates the network needs to maintain the formation movement; and the third type indicates the network needs to change (or transform) the formation.

According to the aforementioned instructions, this section will conduct comparative experiments on the following six situations, and check the trajectory changes of each UAV:

ONLY_KEEP: Without malicious nodes, the ground station sends a commend containing WILL_KEEP.

ONLY_CHANGE: Without malicious nodes, the ground station sends a commend containing WILL_CHANGE.

KEEP_TO_KEEP: The ground station sends a commend containing WILL_KEEP, and the malicious node tampered with the direction and speed of nodes' movement in the packet received.

KEEP_TO_CHANGE: The ground station sends a commend instruction containing WILL_KEEP. The malicious node tampered with WILL_CHANGE, and it randomly modified the distance that the node should move.

CHANGE_TO_CHANGE: The ground station sends the commend instruction with WILL_CHANGE. The malicious node first records the packet's commend information, judges the new formation based on multiple packets' information, and then tampers with the packet's distance information and forwards the packet according to the destination address. The network is modified to an unexpected formation.

CHANGE_TO_KEEP: The ground station sends a commend command containing WILL_CHANGE. The malicious node tampered with WILL_KEEP, and it randomly modified the direction and speed of nodes' movement.

In the packet-tampering experiment in this section, first set UANET's initial formation through NS-3's built-in mobility module. All nodes will maintain the formation movement or modify it to a new formation based on the commend field in the packets. Taking a network with 25 nodes as an example, the UAV network's initial formation is a 5*5 matrix queue. In situation *ONLY_KEEP*, Figure 11 shows the UAV network trajectory after receiving the following commend field (see Table 1).

In the situation *ONLY_CHANGE*, after receiving commend field as Table 2, Figure 12 shows the UAV network trajectory.

If the malicious node receives the commend field with the same content in *ONLY_KEEP*, indicating that the formation is maintained, then according to *KEEP_TO_KEEP*, it will tamper with the direction and speed of the nodes' movement in the packet. Figure 13 compares the UAV network trajectories between the cases *ONLY_KEEP* and *KEEP_TO_KEEP*.

According to *KEEP_TO_CHANGE*, it will change the *WILL_KEEP* in the commend field to *WILL_CHANGE*, and randomly modifies the distance the node should move. Figure 14 compares the UAV network trajectories between cases *ONLY_KEEP* and *KEEP_TO_CHANGE*.

If the malicious node receives the commend field with the same content in *ONLY_CHANGE*, indicating that the formation is maintained, according to *CHANGE_TO_CHANGE*, it will tamper with the packet's distance information. Figure 15 compares the UAV network trajectory between cases *ONLY_CHANGE* and *CHANGE_TO_CHANGE*.

According to *CHANGE_TO_KEEP*, it will change the *WILL_CHANGE* in the commend field to *WILL_KEEP*, and randomly modify the direction and speed of the nodes' movement. Figure 16 compares the UAV network trajectories between cases *ONLY_CHANGE* and *CHANGE_TO_KEEP*.

It is obvious that the malicious node can disrupt UANET's formation and trajectory, and it changes the expected command from the ground control station to the UAV queue, thereby gaining control of the network.

4.3 | Experimental results and analysis

In the SBHA algorithm's simulation experiment for the packet-capture module, after simulating five different network situations and checking the packet loss rate and network throughput, the results show that the SBHA algorithm captures up to 44% of all packets and offers remarkable network throughput reduction.

When no malicious node attacks the network, the packet loss rate is the lowest. At this time, packet loss may be caused by node movement or transmission to unreachable nodes; when malicious nodes perform a traditional black hole attack, the packet loss rate is a 10% higher-than-normal

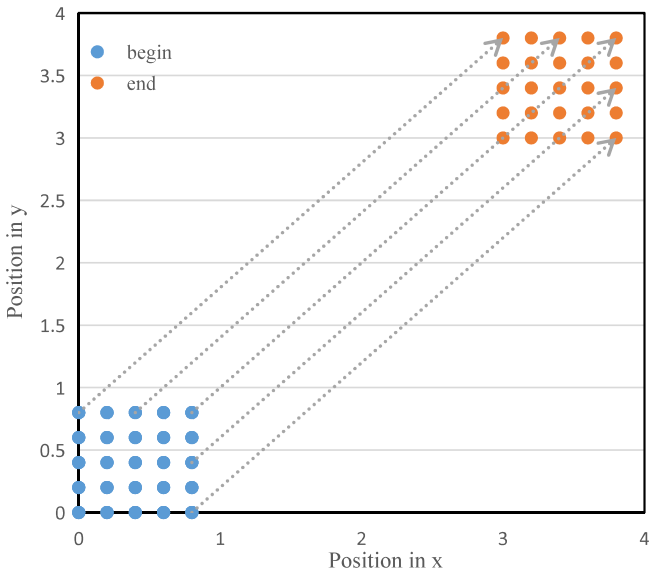

FIGURE 11 UAV network trajectory diagram in the *ONLY_KEEP* situation

TABLE 1 The content of the commend field with *WILL_KEEP*

WILL_KEEP	
1.8	0.6

TABLE 2 Content of the commend field with *WILL_CHANGE*

WILL_CHANGE		
10.1.2.1	104.6	76.8
10.1.2.2	108.6	84.9
10.1.2.3	116.7	87.4
10.1.2.4	119.3	83.5
10.1.2.5	104.2	64.4
⋮	⋮	⋮
10.1.2.24	125.9	52.8
10.1.2.25	94.9	43.3

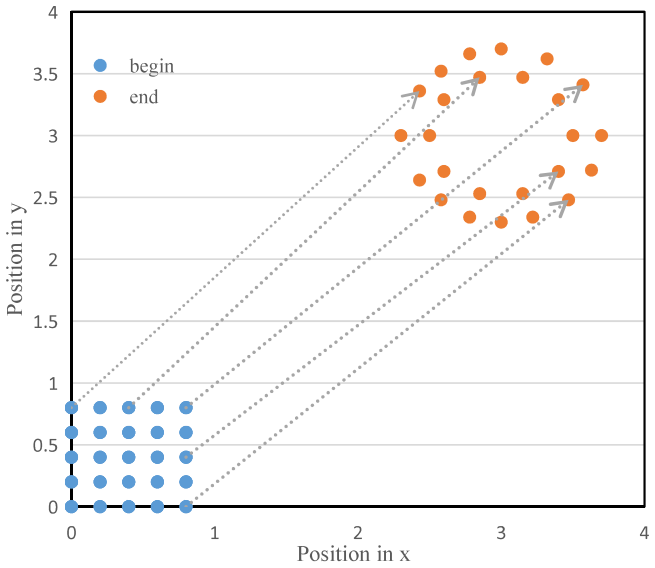

FIGURE 12 UAV network trajectory diagram in the *ONLY_CHANGE* situation

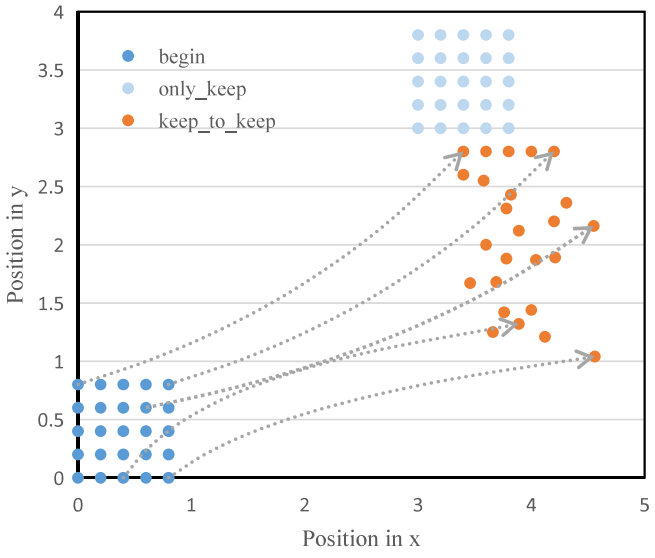

FIGURE 13 Comparing the UAV network trajectories of *ONLY_KEEP* and *KEEP_TO_KEEP*

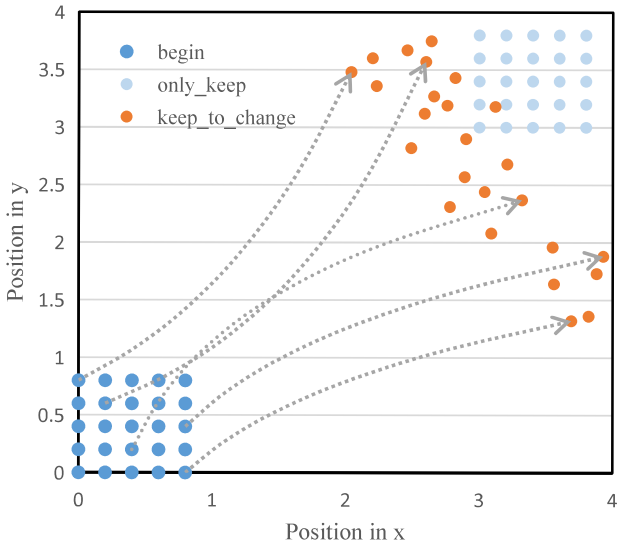

FIGURE 14 Comparing the UAV network trajectories of *ONLY_KEEP* and *KEEP_TO_CHANGE*

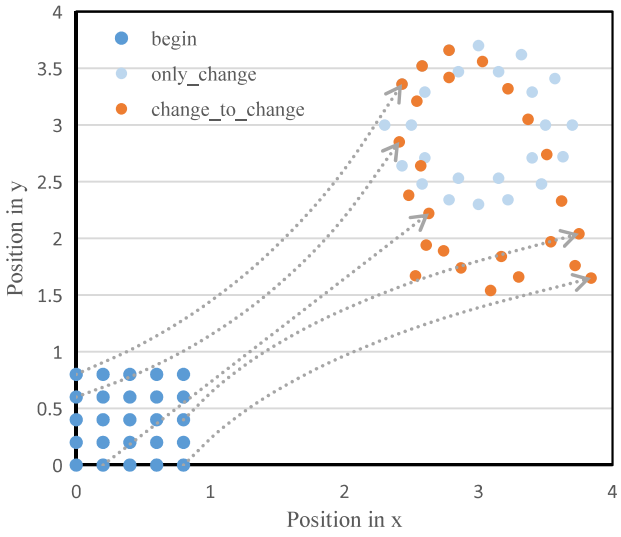

FIGURE 15 Comparing the UAV network trajectories of *ONLY_CHANGE* and *CHANGE_TO_CHANGE*

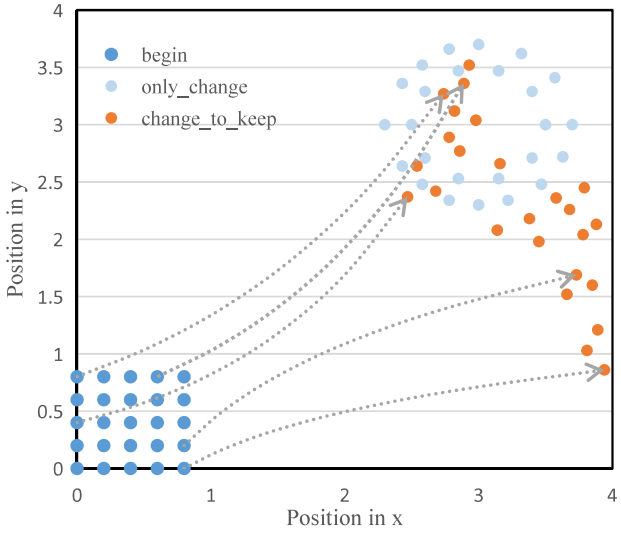

FIGURE 16 Comparing the UAV network trajectories of *ONLY_CHANGE* and *CHANGE_TO_KEEP*

situation. However, after running the prevention program, the packet loss rate almost returned to normal. When a malicious node executes the SBHA algorithm, the packet loss rate increases by 10%–20% compared to normal conditions, and as the network scale increases, the packet loss rate increases more. In addition, after running the prevention program, the packet loss rate shows almost no change. Delay and jitter of the network do not have more changes but can also show the negative impact of UANET by SBHA.

When malicious nodes perform traditional black hole attacks, the throughput is slightly reduced, and after running the trust mechanism scheme, the network throughput increases year-on-year. However, when the malicious node executes the SBHA algorithm, the network throughput can be reduced by up to 90%, and the trust mechanism scheme is still unable to pick up.

In the simulation experiment of tampering with the SBHA algorithm's packets module, it is divided into six situations to show the SBHA algorithm's influence on the UANET's trajectory. In the *ONLY_KEEP* situation, the UAV network maintains the grid formation to fly normally, and in *ONLY_CHANGE*, the UAV network transforms the grid formation into a circle. Meanwhile, *KEEP_TO_KEEP* and *KEEP_TO_CHANGE* are based on *ONLY_KEEP* to make malicious nodes run the SBHA algorithm to tamper with the packets module, respectively. They represent changing the UAVs' direction of movement and state to change the formation. Finally, *CHANGE_TO_CHANGE* and *CHANGE_TO_KEEP* make the malicious node run the SBHA algorithm on the basis of *ONLY_CHANGE* to tamper with the packets module, respectively. They represent changing UAV's distance direction and state to maintain the formation. Figures 10–13 show that the UAVs' movement trajectory has been changed, making the network team form an unpredictable formation except for malicious nodes.

4.4 | The real world UAVs test

We use F450 hexarotor UAV to build the system. The UAVs' experimental function module diagram is shown in Figure 17. The UAVs and the ground station are located in the ad hoc network through the network communication module. After the module receives the ground station instruction

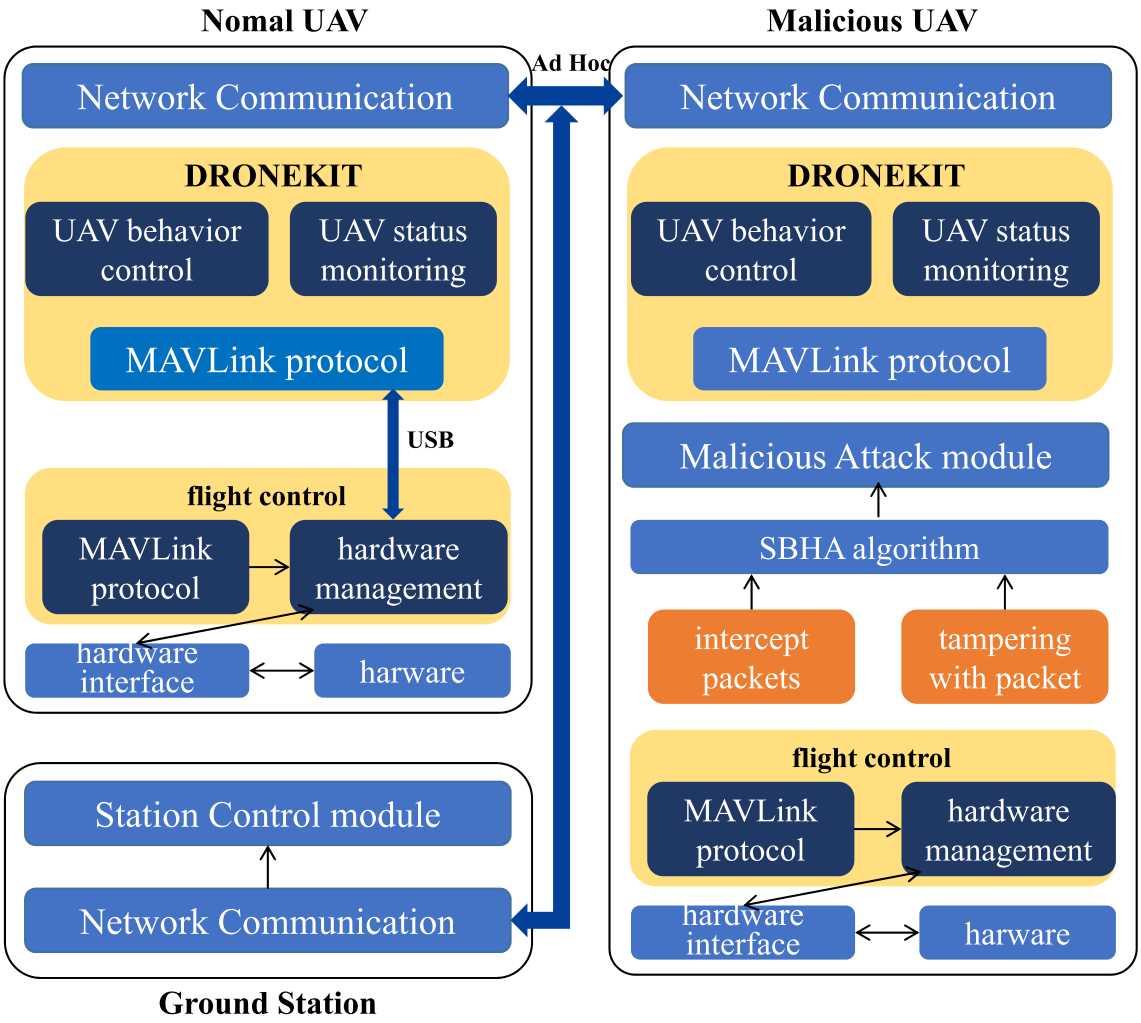

FIGURE 17 UAVs' experimental function module diagram

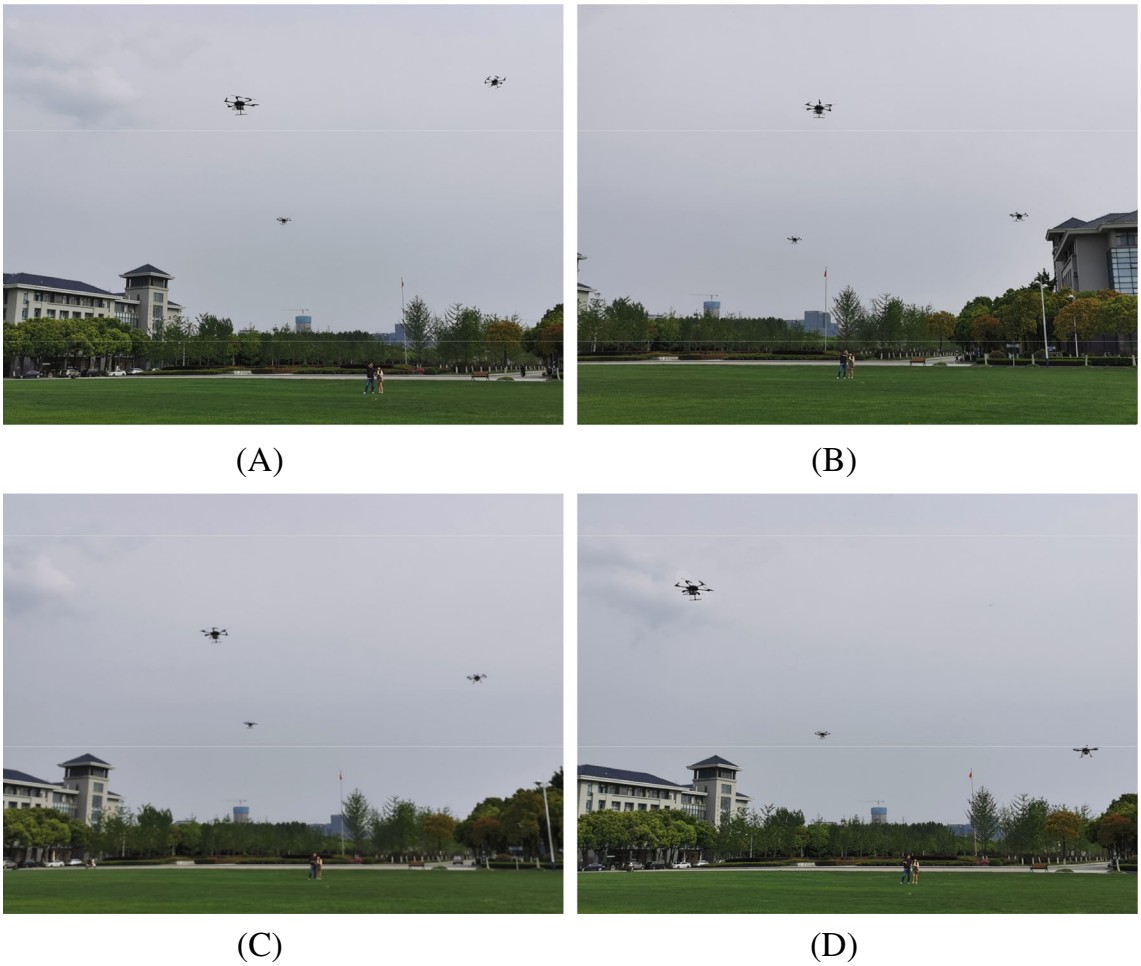

FIGURE 18 The flight change diagram of the test UAVs. (A) Initial hover position. (B) Malicious code running. (C) The formation of UAVs began to change. (D) The distance between UAVs began to increase

in the Raspberry Pi of the UAV, it communicates with the flight control board through the MAVLink protocol to complete the corresponding flight control operations. The system used dronekit software to the MAVLink protocol module, the UAV behavior control module, and the UAV status monitoring module is the flight control module, which is connected to the flight control board hardware management in the Raspberry Pi to control the mobility of UAVs. The ground control module monitors the status information of the UAV group through the feedback of the commands from the UAVs in the network. After completing the networking and connectivity of the UANET, malicious nodes join the network to implement malicious attack modules to verify the SBHA algorithm.

After completing the UAV networking, this article tested SBHA algorithm on multiple UAVs. Figure 18 shows the flight change diagram of the test UAVs.

5 | CONCLUSION

In this article, we compared the importance of topology and traffic centers in forwarding packets. According to the central nodes' two types of characteristics, we proposed a black hole attack algorithm (called SBHA) for UANET based on OLSR. This algorithm is undetectable for existing defense measures, depending on UANET's multi-hop routing and OLSR's known topology. We also optimize the algorithm according to UAV's characteristics, and finally achieve the goal of gaining network control. The simulation results show that SBHA can cause greater damage to UANET compared to a traditional black hole attack, and ordinary defense algorithms cannot reduce SBHA's negative impact on UANET. In future work, we will focus on optimizing SBHA's performance in terms of node energy consumption and the delay of attacking the target node.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China under grants 62172091, 61602112, and 61632008; the International S&T Cooperation Program of China under grant 2015DFA10490; the Jiangsu Provincial Key Laboratory of Network and Information Security under grant BM2003201; the Key Laboratory of Computer Network and Information Integration of the Ministry of Education of China under grant 93K-9; and partially supported by the Collaborative Innovation Center of Novel Software Technology and Industrialization and the Collaborative Innovation Center of Wireless Communications Technology.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Runqun Xiong https://orcid.org/0000-0002-1941-5586
Lan Xiong https://orcid.org/0000-0003-2696-6260
Junzhou Luo https://orcid.org/0000-0001-7518-4367

REFERENCES

1. Clausen TH, Jacquet P. Optimized link state routing protocol (OLSR). RFC 3626; 2003:1-75.
2. Jiang J, Han G. Routing protocols for unmanned aerial vehicles. *IEEE Commun Mag*. 2018;56(1):58-63.
3. Maccari L, Maischberger M, Cigno RL. Where have all the MPRs gone? on the optimal selection of multi-point relays. *Ad Hoc Netw*. 2018;77:69-83.
4. Rosati S, Kruzelecki K, Heitz G, Floreano D, Rimoldi B. Dynamic routing for flying ad hoc networks. *IEEE Trans Veh Technol*. 2016;65(3):1690-1700.
5. Zapata MG, Asokan N. Securing ad hoc routing protocols. Proceedings of the Workshop on Wireless Security; 2002:1-10; ACM, New York, NY.
6. Hu Y, Perrig A, Johnson DB. Ariadne: a secure on-demand routing protocol for ad hoc networks. *Wirel Netw*. 2005;11(1-2):21-38.
7. Al-Karaki JN, Kamal AE. Stimulating node cooperation in mobile ad hoc networks. *Wirel Pers Commun*. 2008;44(2):219-239.
8. Zhao B, Zhao P, Fan P. ePUF: a lightweight double identity verification in IoT. *Tsinghua Sci Technol*. 2020;25(5):625-635.
9. Deng H, Li W, Agrawal DP. Routing security in wireless ad hoc networks. *IEEE Commun Mag*. 2002;40(10):70-75.
10. Perkins CE, Belding-Royer EM, Das SR. Ad hoc on-demand distance vector (AODV) routing. RFC 3561; 2003:1–37.
11. Gerhards-Padilla E, Aschenbruck N, Martini P, Jahnke M, Tölle J. Detecting black hole attacks in tactical MANETs using topology graphs. 32nd IEEE Conference on Local Computer Networks (LCN 2007); 2007:1043-1052; IEEE Computer Society.
12. Er-Rouidi M, Moudni H, Mouncif H, Merbouha A. An energy consumption evaluation of reactive and proactive routing protocols in mobile ad-hoc network. Proceedings of the International Conference on Computer Graphics, Imaging and Visualization; 2016; IEEE.
13. Raffo D, Adjih C, Clausen TH, Mühlethaler P. An advanced signature system for OLSR. Proceedings of the Workshop on Security of Ad Hoc and Sensor Networks; 2004:10-16; ACM, New York, NY.
14. Schweitzer N, Stulman A, Margalit RD, Shabtai A. Contradiction based gray-hole attack minimization for ad-hoc networks. *IEEE Trans Mob Comput*. 2017;16(8):2174-2183.
15. Cai RJ, Li XJ, Chong PHJ. An evolutionary self-cooperative trust scheme against routing disruptions in MANETs. *IEEE Trans Mob Comput*. 2019;18(1):42-55.
16. Zhong X, Chen F, Guan Q, Ji F, Yu H. On the distribution of nodal distances in random wireless ad hoc network with mobile node. *Ad Hoc Netw*. 2020;97:102026.
17. Schweitzer N, Stulman A, Hirst T, Margalit RD, Shabtai A. Network bottlenecks in OLSR based ad-hoc networks. *Ad Hoc Netw*. 2019;88:36-54.
18. Kimura N, Yoshiura N. Construction and verification of mobile ad hoc network protocols. In: Wang G, Ray I, Feng D, Rajarajan M, eds. *Cyberspace Safety and Security*. Springer; 2013:198-212.
19. Huan Y. Reliable data storage in heterogeneous wireless sensor networks by jointly optimizing routing and storage node deployment. *Tsinghua Sci Technol*. 2019;26(2):230-238.
20. Huo Y, Fan J, Wen Y, Li R. A cross-layer cooperative jamming scheme for social Internet of Things. *Tsinghua Sci Technol*. 2020;26(4):523-535.
21. Demir U, Tapparello C, Heinzelman WB. Maintaining connectivity in ad hoc networks through WiFi direct. Proceedings of the IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems; 2017:308-312; IEEE.
22. NS-3. https://www.nsnam.org/

How to cite this article: Xiong R, Xiong L, Shan F, Luo J. SBHA: An undetectable black hole attack on UANET in the sky. *Concurrency Computat Pract Exper*. 2021;e6700. doi: 10.1002/cpe.6700