Mitigating Routing Misbehavior in the Internet of Drones Environment

Cong Pu and Pingping Zhu
Dept. of CSEE, Marshall University, Huntington, WV 25755, USA. Email: cong.pu@ieee.org, zhup@marshall.edu

***Abstract*—Despite initially made for military purposes, drones have presented themselves to consumers, and the drone industry is expected to witness a significant growth during the forecast period. As the number of drones in the sky keeps growing, a fleet of drones and stationary zone service providers (ZSPs) can form an airborne network which is termed the Internet of Drones (IoD). In order to achieve the objectives of efficient information sharing and superior team performance, routing protocol plays a vital role for reliable communication in the IoD. However, malicious drones may strategically drop any received packets, and traditional mitigation techniques designed specially for mobile/vehicular ad hoc networks are unable to be directly applied in the IoD as a consequence of the intermittent connectivity between drones. In this paper, we propose a distributed countermeasure, also called $Counter^{Romir}$, to detect and mitigate routing misbehavior in the IoD. In $Counter^{Romir}$, a drone keeps the previous signed communication invoice and shares it with the next-hop drone so that the next-hop drone can detect whether the drone has dropped any packets or not. In consideration of a malicious drone likely misstating its communication invoice to avoid detection, each drone saves and sends a small number of past communication invoices to the ZSP which can detect the misstating drone. We develop a comprehensive simulation framework and conduct extensive simulation experiments using OMNeT++ for performance evaluation and analysis. After comparing with prior schemes, we come to the conclusion that $Counter^{Romir}$ can provide admirable performance in terms of detection rate, packet delivery ratio, miss/error detection rate, and the number of dropped packets, indicating an applicable approach against routing misbehavior in the IoD.**

***Index Terms*—Drones, Flying Ad Hoc Networks, Internet of Drones, Routing Misbehavior, Detection and Mitigation**

I. Introduction

The initial use of drones was strike weapons as remotely-guided aerial missile deployers. Today, drones have discovered a variety of applications for civilian use such as goods delivery, aerial surveillance, search and rescue, and combating coronavirus (COVID-19) pandemic [1]. According to ‘‘Drone Market Report 2020’’, the drone industry is expected to grow to $43 billion by 2025 [2]. As more and more drones invade and occupy our airspace, a fleet of drones and stationary zone service providers (ZSPs) can form an airborne network, which is termed the Internet of Drones (IoD) [3], to carry out a range of challenging tasks. Instead of solely depending on fixed infrastructure, the IoD exploits the intermittent connectivity between drones for the dissemination of information in the highly dynamic environment. In addition, drones may occasionally use ZSPs to connect to the Internet, gathering up-to-date information for their specific tasks [4]. However, not every drone has a direct connection with the ZSP due to the deployment costs of ZSPs. Thus, store-carry-and-forward mechanism and delay tolerant networking technique [5] can be adopted in the IoD, where a drone receives some packets, stores and carries them while flying, and finally forwards them to the next-hop drone or nearby ZSP.

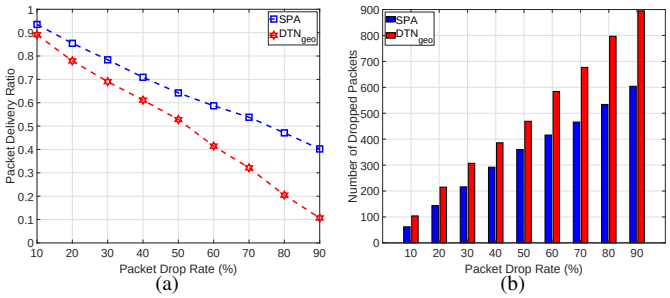

Fig. 1. Preliminary experiments using OMNeT++: the impact of routing misbehavior when SPA and DTN$_{geo}$ are simulated as the routing protocols.

In order to achieve the objectives of efficient information sharing and superior team performance, routing protocol plays a vital role for reliable communication in the IoD [6]. However, a malicious drone may strategically misbehave by dropping the received packets because either it is willing to save energy power or it is launching attacks. Routing misbehavior can extensively degrade the performance of ad hoc networks, e.g., causing reduction in the packet delivery ratio and increasement in the number of dropped packets [7]. To further demonstrate the impact of routing misbehavior, we carry out preliminary experiments in OMNeT++ [8] using two IoD routing protocols, stochastic packet forwarding (SPA) [9] and motion-driven packet forwarding (DTN$_{geo}$) [10]. SPA is a stochastic packet forwarding algorithm where the receiving drone is stochastically selected based on the calculated forwarding probability. DTN$_{geo}$ is a shortest path forwarding algorithm where the source drone tries to find the Dijkstra shortest path to send the packets. As shown in Fig. 1(a), when the packet drop rate increases, the packet delivery ratio of SPA and DTN$_{geo}$ continuously decreases. When the packet drop rate reaches 90%, only approximate 40% and 10% packets can be delivered by SPA and DTN$_{geo}$, respectively. In addition, the number of dropped packets for SPA as well as DTN$_{geo}$ is shown in Fig. 1(b). A survey paper [11] discusses the existing routing algorithms and their potential security attacks in the IoD, but it fails to study and investigate the impact of routing misbehavior and propose the corresponding countermeasure. Hence, it is essential to detect packet dropping attack and mitigate potential routing misbehavior in the IoD.

Over the last decade, routing misbehavior and its countermeasures have been investigated in various environments, such

as wireless ad hoc networks [12], mobile ad hoc networks [13], vehicular ad hoc networks [14], etc. In summary, various countermeasures can be briefly categorized into monitoring-based, acknowledgment-based, bait-based, and cryptography-based approaches [15]. Those approaches have some merits in the mitigation of routing misbehavior. However, they fail to consider the high mobility and the intermittent connectivity, thus cannot be straightly adopted in the IoD.

In this paper, we focus our attention on routing misbehavior and propose a distributed countermeasure in the IoD. Our major contribution can be briefly summarized below:

1) We propose a distributed countermeasure, also called $Counter^{Romir}$, to detect and mitigate routing misbehavior in the IoD. In $Counter^{Romir}$, a drone keeps the previous signed communication invoice and shares it with the next-hop drone so that the next-hop drone can detect whether the drone has dropped any packets.
2) In consideration of the malicious drone likely misstating its communication invoices to avoid detection, we propose that each drone saves and sends a small number of past communication invoices to the ZSP which can detect the misstating drone.
3) We develop a comprehensive simulation framework and conduct extensive simulation experiments using OMNeT++ for performance evaluation and analysis. We also implement prior schemes such as SCAD [16] and EYES [15] for performance comparison.

Based on extensive simulation experiments, we reach the conclusion that $Counter^{Romir}$ is an efficient approach to mitigate routing misbehavior in the IoD. In the following, we first review the existing work in Section II. Then we propose the routing misbehavior countermeasure in Section III. Experiments and their analysis are provided in Section IV. Finally, we conclude the paper in Section V.

II. Related Work

Routing misbehavior was first addressed in [17], where a watchdog scheme is deployed to implicitly monitor the activity of next-hop node and determine whether it forwards the recently received packets. And on this basis some researchers further investigated the implicit monitoring technique and proposed various variants [15], [18] to detect and mitigate routing misbehavior. Nevertheless, implicit monitoring completely depends on stable connectivity between sender and receiver, which is an extremely demanding condition in the IoD. This is because drones have high mobility, and the packet sender might not still be within the communication range of packet receiver to monitor the follow-up operations.

Acknowledgment-based approach for the detection of routing misbehavior became another mainstream after 2ACK technique was proposed in [19]. In 2ACK, a two-hop acknowledgment packet is replied to the opposite direction of data traffic to detect misbehaving links or nodes. Some follow-up examinations such as single-checkpoint scheme [16] and EAACK+RSA [20] have adopted explicit acknowledgment approach to detect and mitigate packet dropping attack. However, the acknowledgment-based approach is not applicable in the IoD because the reverse path might not exist anymore when the acknowledgment packet is being forwarded back.

Another branch of routing misbehavior study is called bait-based approach, where the fictitious information is fabricated to lure adversaries to launch attacks. The authors in [7] and [21] desire to use fake route request packet to expose the potential adversaries in the network. If any suspicious node replies the fake request packet, it will be considered an adversary and a follow-up mitigation process will be initiated. Although bait-based approaches can achieve low false negative rate, the fake request packets could easily get lost during the transmission due to the high mobility in the IoD.

To address the routing misbehavior in the IoD, the authors in [22] design a trust management scheme to distinguish between legitimate and malicious forwarding behaviors. A fuzzy trust scheme and decay function are proposed to examine node's trustworthiness and converge trust, reward, and punishment values, respectively. Using the trust value, a cluster head is selected for both intra and inter-cluster communication. However, the trust evaluation process still relies on neighbor monitoring and the cluster-head selection procedure incurs additional communication overhead. To defend against jamming attack, federated learning-based cognitive detection [23] and cross-layer anti-jamming routing [24] are proposed. However, those emerging approaches are not really applicable to detect packet dropping attack in the IoD.

Our approach $Counter^{Romir}$ borrows the idea of store-carry-and-forward mechanism and delay tolerant networking technique to address the challenging issue of intermittent connectivity in the IoD, where a drone keeps the previous signed communication invoices and shares them with the next-hop drone or nearby ZSP to detect the routing misbehavior or misstating drones. In addition, $Counter^{Romir}$ is a network-layer approach which can be implemented as an add-on to existing routing protocols (e.g., SPA, DTN_{geo}, etc.) in the IoD. Finally, to the best of our knowledge, there is no existing distributed countermeasure against routing misbehavior in the IoD, and $Counter^{Romir}$ will bridge this research gap.

III. The Proposed Countermeasure

A. System and Adversary Models

A system model is shown in Fig. 2, where drone-to-drone and drone-to-zsp communications are supported by IEEE 802.11p standard [25]. We consider a general scenario that a fleet of drones (denoted as N_i) is deployed in an area for a mission, e.g., enforcing stay-at-home order during the COVID-19 pandemic. Drones can communicate with each other (i.e., drone-to-drone communications) to transfer data and coordinate decision making. To deliver data towards destination, a stochastic packet forwarding [9] or a shortest path forwarding [10] technique can be deployed. However, due to the high mobility of drones, the communication link between drones is not stable and the end-to-end routing path might not always exist between source and destination. Thus, store-carry-and-forward mechanism can be leveraged in the IoD, where a

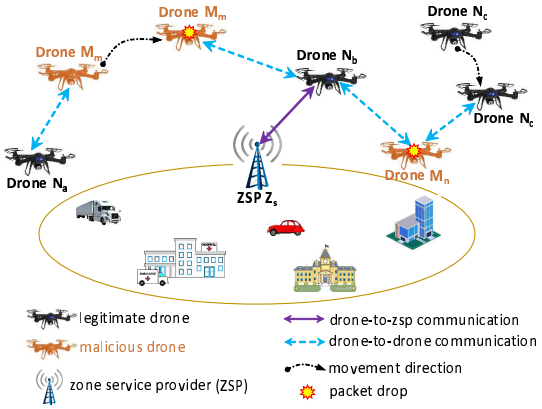

Fig. 2. System model.

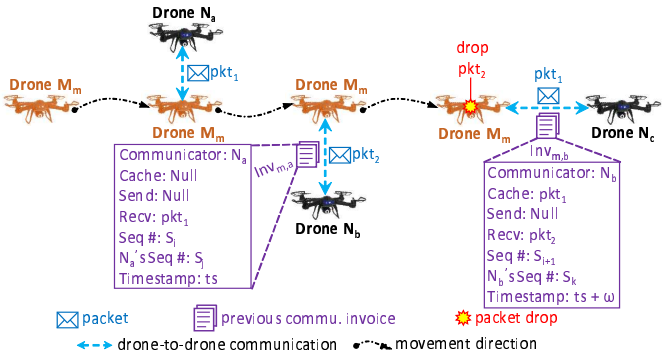

Fig. 3. Example of packet dropping detection.

drone receives some data, stores and carries them while flying, and finally sends them to the next-hop drone or nearby ZSP. In addition, ZSPs located in the area can provide Internet connectivity to drones (e.g., drone-to-zsp communications), but they are not widely available due to the deployment costs. Therefore, not every drone has a direct connectivity with the ZSP. However, we assume that a drone will meet a ZSP sooner or later. We also assume that a public-key cryptography [26], [27] is available in the IoD. In IBE-Lite, an arbitrary string is used to generate a public key separately from the secret key, and a trusted certificate authority is responsible for identity verification and secret key derivation. A drone might fly to a neglected area, where an adversary can capture the drone using the ''anti-drone-gun'', and compromise and send it back to the mission area for malicious purposes. The major objective of malicious drone (denoted as M_i) is to degrade the network performance by strategically dropping the packets. A small number of malicious drones might collude together to drop the packets without being detected. However, the collusive packet dropping attack is out of the scope of this paper.

B. Overview of $Counter^{Romir}$

In $Counter^{Romir}$, when two drones communicate to exchange data, they create a time-stamped communication invoice that itemizes and records communication details relating to them. In order to validate the authenticity and integrity of a communication invoice, both of drones will sign the communication invoice. A drone needs to keep the invoice of previous communication and shares it with the next-hop drone so that the next-hop drone can examine the communication invoice and detect whether the drone has dropped any data. A malicious drone likely misstates its communication invoice to avoid detection. However, misstating will cause the inconsistency in the communication invoices issued by the malicious drone. Thus, each drone is required to save and send a small number of past communication invoices to the ZSP which can detect the misstating drone. More details about $Counter^{Romir}$ are provided in the following.

C. $Counter^{Romir}$: Routing Misbehavior Countermeasure

First, when two drones come into each other's communication range, they exchange the packets to be sent to the next-hop drone and create a communication invoice. The communication invoice includes the communicators' ID, the timestamp of communication, the unique communication sequence number assigned by each of them, what packets are in their caches before the communication, what packets they receive and send during the communication, and their digital signatures. Here, the communication invoice is a certified record that contains all communication related information of two drones. A drone needs to keep the previous communication invoice, and share it with the next-hop drone before they exchange any packet. In addition, the drone is required to send the vector of packets in its cache to the next-hop drone. After receiving the communication invoice and the vector of cached packets, the next-hop drone examines the information and determines whether the drone has dropped any packet. If the next-hop drone does not detect any packet dropping, it will exchange the packets with the drone as normal. Otherwise, the next-hop drone will quit sending the packets to the suspected drone.

As shown by the example in Fig. 3, malicious drone M_m first communicates with drone N_a and receives the packet pkt_1. We assume that M_m did not store any packet in the cache before the communication. After the packet exchange, M_m creates the invoice of communication between itself and N_a, denoted as $Inv_{m,a}$, which will be shared with the next-hop drone (e.g., N_b). The communication invoice $Inv_{m,a}$ is represented as follows

$$Inv_{m,a} = [M_m, N_a, TS, Seq_m, Seq_a, Ca_m, Rec_m, Sen_m, Sig_m, Sig_a]. \tag{1}$$

Here, Seq_m and Seq_a are the unique communication sequence number assigned by M_m and N_a, respectively. TS is the timestamp of communication, Ca_m is the vector of packets cached by M_m before the communication, Rec_m is the packets received by M_m, and Sen_m is the packets sent by M_m. Sig_m and Sig_a are the digital signature created by M_m and N_a, respectively, and can be represented as

$$\begin{aligned} Sig_m &= DSA\big(H(M_m|N_a|TS|Seq_m|Seq_a|Ca_m|Rec_m|Sen_m), PR_m\big), \\ Sig_a &= DSA\big(H(M_a|N_m|TS|Seq_a|Seq_m|Ca_a|Rec_a|Sen_a), PR_a\big), \end{aligned} \tag{2}$$

where $DSA(\cdot)$ is the digital signature algorithm, $H(\cdot)$ is the secure one-way hash function, and $|$ is the concatenation operation. PR_m and PR_a are the private key of M_m and N_a, respectively. The communication invoice created by N_a (e.g.,

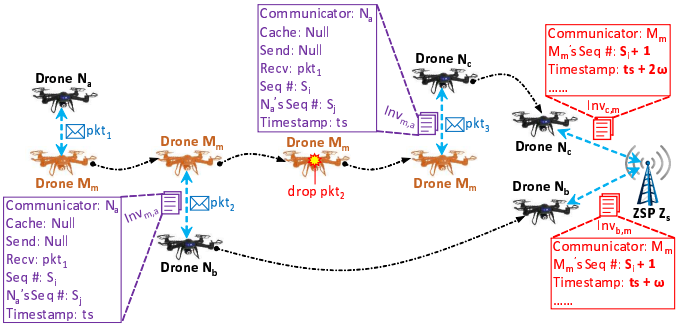

Fig. 4. Example of communication invoice misstating detection.

$Inv_{a,m}$) is similar to $Inv_{m,a}$ except that Ca_m, Rec_m, and Sen_m are replaced by Ca_a, Rec_a, and Sen_a, respectively.

When M_m meets and plans to exchange the packets with N_b, it first sends the vector of its cached packets and the previous communication invoice $Inv_{m,a}$ to N_b. Since N_b examines the information (e.g., M_m received the packet pkt_1; and the packet pkt_1 is in M_m's cache.) and does not detect any packet dropping, it sends the packet pkt_2 to M_m. When M_m meets with the next-hop drone N_c, it drops the packet pkt_2. However, this packet dropping can be easily detected by N_c. This is because after N_c receives and examines the communication invoice $Inv_{m,b}$ and the vector of packets cached by M_m, it finds that the packet pkt_2 was received by M_m but it is not stored in the cache. As a result, N_c suspects the packet dropping of M_m and quits sending packets to M_m.

Second, a malicious drone may share the incorrect communication invoice to cover up its packet dropping activity and avoid being detected. However, misstating will bring inconsistent communication invoices to the network, and the ZSP can use these inconsistent communication invoices to detect the misstating activity of malicious drone. In this paper, we propose that a drone will incrementally assign a unique communication sequence number to each communication, and the same sequence number will not be used twice. For example, the first communication has the sequence number 1, and the second communication has 2 as the sequence number, and so on. The rationale behind this design is that the 32-bit communication sequence number space contains 2^{32} possible numbers, which is assumed to be large enough. The basic idea of detecting the activity of misstating is that each drone saves a small number of invoices of communications with other drones and sends them to ZSPs for verification. Thus, there is a high chance that the ZSP receives the inconsistent communication invoices from two different drones, and the misstating activity of malicious drone can be detected.

As shown by the example in Fig. 4, malicious drone M_m first communicates with drone N_a and N_b, and receives the packet pkt_1 and pkt_2, respectively. Before communicating with drone N_c, M_m drops the packet pkt_2. In order to hide its packet dropping activity, M_m intentionally shares the invoice of communication with N_a, $Inv_{m,a}$, with N_c, instead of the invoice of communication with N_b, $Inv_{m,b}$. In other words, M_m pretends that it did not communicate with N_b before. Under the circumstances, N_c cannot detect the packet dropping activity of M_m based on the received

Algorithm 1: Routing Misbehavior Countermeasure

Input: $Inv_{m,a}$, Ca_m, $Inv_{b,m}$, $Inv_{c,m}$

```
/* drone detects packet dropping attack                    */
Function DroneDetect(Inv_{m,a}, Ca_m):
    /* Inv_{m,a}[Ca_m] is the vector of cached
       packets at the beginning of previous
       communication; Ca_m is the vector of
       cached packets at the beginning of
       current communication.                               */
    /* pkt indicates the packet.                            */
    if pkt ∈ (Inv_{m,a}[Ca_m] ∪ Inv_{m,a}[Rec_m]) and pkt ∉ Ca_m
      and pkt ∉ Inv_{m,a}[Sen_m] then
        detect packet dropping misbehavior;
    else
        exchange packets;
    end
/* ZSP detects commu. invoice misstating                   */
Function ZSPDetect(Inv_{b,m}, Inv_{c,m}):
    if Inv_{b,m}[TS] < Inv_{c,m}[TS] then
        if Inv_{b,m}[Seq_m] ≥ Inv_{c,m}[Seq_m] then
            detect communication invoice misstating;
            broadcast Alarm packet;
        end
    end
    if Inv_{b,m}[TS] > Inv_{c,m}[TS] then
        if Inv_{b,m}[Seq_m] ≤ Inv_{c,m}[Seq_m] then
            detect communication invoice misstating;
            broadcast Alarm packet;
        end
    end
```

communication invoice $Inv_{m,a}$ and the vector of packets cached by M_m. However, since M_m declares that N_a was the previous communicating drone, the communication sequence number in $Inv_{m,a}$ increased by one will become the sequence number assigned for the communication with N_c. Simply put, M_m will have to assign the same sequence number for the communication with N_b and N_c at two different times, $ts + \omega$ and $ts + 2\omega$. However, this violates the pre-defined communication sequence number policy that a drone will incrementally assign a unique communication sequence number to each communication, and the same sequence number will not be used twice. When N_b and N_c share their past communication invoices (e.g., $Inv_{c,m}$ and $Inv_{b,m}$) with the ZSP, the ZSP can easily detect the misstating activity of M_m. After detecting the activity of misstating, the ZSP will broadcast an *Alarm* packet in the network so that M_m can be added in the blacklist and all other drones will not send any packet to M_m. The pseudocode of $Counter^{Romir}$ is described in Algorithm 1.

IV. Performance Evaluation

We conduct extensive simulation experiments using OMNeT++ [8] to evaluate the performance of $Counter^{Romir}$. In the customized simulation framework, 25 legitimate and 5 malicious drones are deployed in a 100×100 network area. The random waypoint mobility model is adopted in the framework, where each drone moves with a constant speed of 15 meter/sec. The radio communication range is set to 12.59 meters, and the radio's data rate is 250 Kbps. Since the size of network is small, a shorter communication range is being adopted. This network setting is similar as the one with a large network size and a longer communication range. The

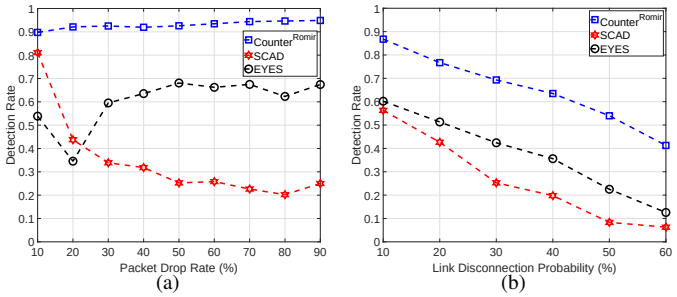

Fig. 5. The performance of detection rate against the packet drop rate and link disconnection probability.

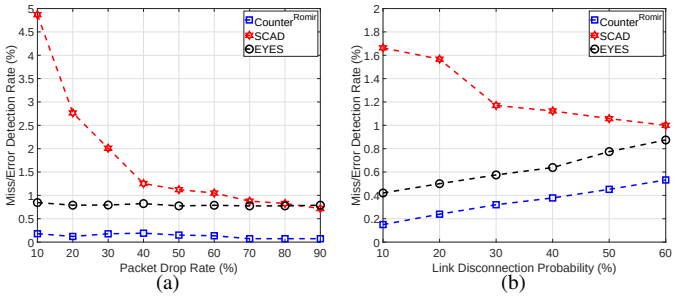

Fig. 6. The performance of miss/error detection rate against the packet drop rate and link disconnection probability.

packet rate is 0.2 pkt/sec and the data payload is 127 bytes. In addition, the wireless channel is assumed to be ideal in the network, where all drones at a certain distance from a transmitter get the exact same signal strength and all links are perfectly bidirectional. The length of simulation is 2000 seconds. For each simulation scenario, the experiment is repeated 5 times with different simulation seed to obtain the steady performance result. We measure the performance in terms of detection rate, miss/error detection rate, packet delivery ratio, and the number of dropped packets. We also compare $Counter^{Romir}$ with SCAD [16] and EYES [15]. In SCAD, the source drone randomly selects a checkpoint drone to detect the packet dropping. If an intermediate drone does not receive the required number of Ack packets, it suspects the next-hop drone located in the path as a malicious drone and generates an *Alarm* packet to report the packet dropping activity. In EYES, each drone monitors the forwarding operations of next-hop drone to detect any packet dropping activity.

In Fig. 5, we measure the detection rate by varying the packet drop rate and link disconnection probability. As shown in Fig. 5(a), the detection rate of $Counter^{Romir}$ is maintained above 90% as the packet drop rate is increased from 10% to 90%. As the packet drop rate increases, more packets would be dropped by the malicious drone. At the same time, more packet dropping activities can also be detected by $Counter^{Romir}$. As a result, a higher detection rate is observed. The detection rate of SCAD decreases as the packet drop rate increases. This is because the unstable links cause the losses of *Alarm* packets and the packet dropping activities of malicious drone cannot be detected. EYES shows a higher detection rate than SCAD because each drone monitors the forwarding operation of next-hop drone and can detect the packet dropping activity if the next-hop drone refuses to forward the packet within a timeout

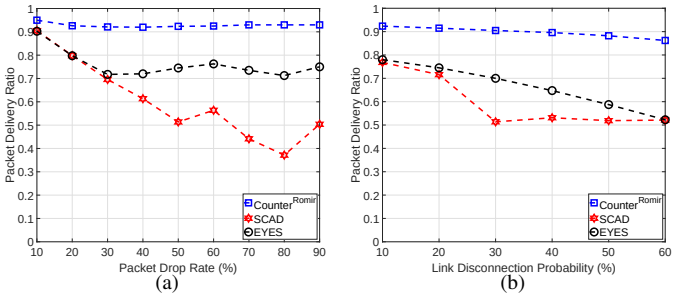

Fig. 7. The performance of packet delivery ratio against the packet drop rate and link disconnection probability.

period. In Fig. 5(b), as the link disconnection probability increases, the overall detection rate of three schemes decrease. This is because the communication links become less stable with a larger link disconnection probability, and a less number of packet dropping activities can be detected. For example, in $Counter^{Romir}$, the communication invoice packet might get lost during the transmission because of unstable communication link. Even though the detection rate is decreasing, $Counter^{Romir}$ still outperforms SCAD and EYES.

In Fig. 6, we measure the miss/error detection rate by changing the packet drop rate and the link disconnection probability. $Counter^{Romir}$ will experience miss detection when the communication invoice packet is lost during the transmission, whereas SCAD and EYES will have error detection when they incorrectly suspect the legitimate drones for packet dropping activities. As shown in Fig. 6(a), the miss detection rate of $Counter^{Romir}$ is as low as 3%, while EYES keeps the error detection rate at 10%. Since $Counter^{Romir}$ stores the communication invoice while flying and forwards it to the next-hop drone when they contact, a lower miss detection rate is obtained. However, there are a few communication invoice packets that could get lost during the transmission, which causes the miss detection. The error detection rate of SCAD significantly decreases as the packet drop rate increases. This is because more packets would be dropped with a larger packet drop rate, but they cannot be detected by SCAD. In Fig. 6(b), the miss detection rate of $Counter^{Romir}$ and the error detection rate of EYES increase as the link disconnection probability increases. For $Counter^{Romir}$, more communication invoice packets will get lost during the transmission, thus a higher miss detection rate is obtained. In EYES, the neighbor drone cannot monitor the forwarding operation of next-hop drone due to unstable communication link, which causes the error detection rate increase.

The performance of packet delivery ratio against the packet drop rate and the link disconnection probability is measured in Fig. 7. In Fig. 7(a), the packet delivery ratio of $Counter^{Romir}$ is maintained above 90% when the packet drop rate increases from 10% to 90%. SCAD shows the lowest packet delivery ratio because it cannot detect enough packet dropping activities and isolate the malicious drone from the network quickly. Since EYES has a higher detection rate than SCAD, more packet dropping activities can be detected and the malicious drone can be isolated more quickly. As a result, a higher

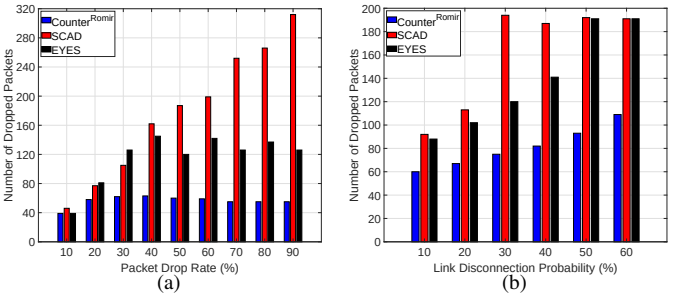

Fig. 8. The performance of the number of dropped packets against the packet drop rate and link disconnection probability.

packet delivery ratio is observed by EYES. As shown in Fig. 7(b), when the link disconnection probability increases, the packet delivery ratio of all three schemes decrease. However, $Counter^{Romir}$ still outperforms SCAD and EYES, and provides the highest packet delivery ratio.

The performance of the number of dropped packets with varying packet drop rate and link disconnection probability is measured in Fig. 8. In Fig. 8(a), the number of dropped packets in SCAD increases linearly as the packet drop rate increases. The rationale is that more packets will be dropped with a larger packet drop rate, and those packet dropping activities cannot be detected. EYES experiences an increasing and decreasing number of dropped packets. This is because EYES can detect enough packet dropping activities and isolate the malicious drone from the network, a less number of dropped packets is observed later. $Counter^{Romir}$ shows the lowest number of dropped packets. In Fig. 8(b), the number of dropped packets increases as the link disconnection probability increases. With a larger link disconnection probability, the communication links become less stable. As a result, less number of packet dropping can be detected and it takes a longer time to isolate the malicious drone from the network. Thus, a larger number of dropped packets is observed for all three schemes.

V. Conclusion

In this paper, we proposed a distributed countermeasure ($Counter^{Romir}$) to detect and mitigate routing misbehavior in the IoD. The basic idea of $Counter^{Romir}$ is that a drone keeps the previous signed communication invoice and shares it with the next-hop drone so that the next-hop drone can detect whether the drone has dropped any packet. To detect the malicious drone that misstates its communication invoice to avoid detection, we proposed that each drone saves and sends a small number of past communication invoices to the ZSP which can detect the misstating drone. Through experimental study, we found that $Counter^{Romir}$ can achieve 90% detection rate as well as maintain the packet delivery ratio above 90%. Moreover, a lower miss/error detection rate is obtained in $Counter^{Romir}$ compared to other schemes. In summary, $Counter^{Romir}$ is an applicable approach against routing misbehavior in the IoD.

References

[1] C. Pu, I. Ahmed, E. Allen, and K. Choo, "A Stochastic Packet Forwarding Algorithm in Flying Ad Hoc Networks: Design, Analysis, and Evaluation," *IEEE Access*, vol. 9, pp. 162 614–162 632, 2021.

[2] *Drone Market Report 2020*, 2020, https://droneii.com/product/drone-market-report-2020-2025.

[3] P. Boccadoro, D. Striccoli, and L. Grieco, "An extensive survey on the Internet of Drones," *Ad Hoc Networks*, vol. 122, p. 102600, 2021.

[4] C. Pu and L. Carpenter, "Psched: A Priority-Based Service Scheduling Scheme for the Internet of Drones," *IEEE Systems Journal*, vol. 15, no. 3, pp. 4230–4239, 2021.

[5] C. Pu and P. Zhu, "Defending against Flooding Attacks in the Internet of Drones Environment," in *Proc. IEEE GLOBECOM*, 2021, pp. 1–6.

[6] D. Lakew, U. Sa'ad, N. Dao, W. Na, and S. Cho, "Routing in Flying Ad Hoc Networks: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 1071–1120, 2020.

[7] C. Pu, S. Lim, J. Chae, and B. Jung, "Active detection in mitigating routing misbehavior for MANETs," *Wireless Networks*, vol. 25, no. 4, pp. 1669–1683, 2019.

[8] A. Varga, *OMNeT++*, 2014, http://www.omnetpp.org/.

[9] C. Pu, "Stochastic Packet Forwarding Algorithm in Flying Ad Hoc Networks," in *Proc. IEEE MILCOM*, 2019, pp. 490–495.

[10] M. Asadpour, K. Hummel, D. Giustiniano, and S. Draskovic, "Route or Carry: Motion-Driven Packet Forwarding in Micro Aerial Vehicle Networks," *IEEE Trans. Mobile Comput.*, vol. 16, no. 3, pp. 843–856, 2017.

[11] O. Oubbati, M. Atiquzzaman, P. Lorenz, and S. M. M. Tareque, "Routing in Flying Ad Hoc Networks: Survey, Constraints, and Future Challenge Perspectives," *IEEE Access*, vol. 7, pp. 81 057–81 105, 2019.

[12] K. Khan, A. Mehmood, S. Khan, M. Khan, Z. Iqbal, and W. Mashwani, "A survey on intrusion detection and prevention in wireless ad-hoc networks," *Journal of Systems Architecture*, vol. 105, p. 101701, 2020.

[13] G. Liu, Z. Yan, and W. Pedrycz, "Data collection for attack detection and security measurement in mobile ad hoc networks: A survey," *Journal of Network and Computer Applications*, vol. 105, pp. 105–122, 2018.

[14] A. Malhi, S. Batra, and H. Pannu, "Security of vehicular ad-hoc networks: A comprehensive survey," *Computers & Security*, vol. 89, p. 101664, 2020.

[15] C. Pu, S. Lim, B. Jung, and J. Chae, "EYES: Mitigating forwarding misbehavior in energy harvesting motivated networks," *Computer Communications*, vol. 124, pp. 17–30, 2018.

[16] C. Pu and S. Lim, "A Light-Weight Countermeasure to Forwarding Misbehavior in Wireless Sensor Networks: Design, Analysis, and Evaluation," *IEEE Systems Journal*, vol. 12, no. 1, pp. 834–842, 2018.

[17] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," in *Proc. IEEE MobiCom*, 2000, pp. 255–265.

[18] M. Rmayti, R. Khatoun, Y. Begriche, L. Khoukhi, and D. Gaiti, "A stochastic approach for packet dropping attacks detection in mobile Ad hoc networks," *Computer Networks*, vol. 121, pp. 53–64, 2017.

[19] K. Liu, J. Deng, P. Varshney, and K. Balakrishnan, "An Acknowledgment-Based Approach for the Detection of Routing Misbehavior in MANETs," *IEEE Transactions on Mobile Computing*, vol. 6, no. 5, pp. 536–550, 2007.

[20] R. Thiyagarajan and B. Priya, "An enhancement of EAACK using P2P ACK and RSA public key cryptography," *Measurement*, vol. 136, pp. 116–121, 2019.

[21] R. Jhaveri, A. Desai, A. Patel, and Y. Zhong, "A Sequence Number Prediction Based Bait Detection Scheme to Mitigate Sequence Number Attacks in MANETs," *Security and Communication Networks*, p. 13, 2018.

[22] K. Singh and A. Verma, "TBCS: A Trust Based Clustering Scheme for Secure Communication in Flying Ad-Hoc Networks," *Wireless Personal Communications*, vol. 114, no. 4, pp. 3173–3196, 2020.

[23] N. Mowla, N. Tran, I. Doh, and K. Chae, "Federated Learning-Based Cognitive Detection of Jamming Attack in Flying Ad-Hoc Network," *IEEE Access*, vol. 8, pp. 4338–4350, 2019.

[24] X. Fan, J. Lin, C. Zhang, and S. Zhang, "A Cross-Layer Anti-Jamming Routing Protocol for FANETS," in *Proc. IEEE ICCC*, 2018, pp. 301–305.

[25] C. Pu, "A Reinforcement Learning Based Service Scheduling Algorithm for Internet of Drones," in *Proc. IEEE ICC Workshop*, 2022, pp. 1–6.

[26] Y. Wu, H. Dai, H. Wang, and K. Choo, "Blockchain-Based Privacy Preservation for 5G-Enabled Drone Communications," *IEEE Network*, vol. 35, no. 1, pp. 50–56, 2021.

[27] C. Pu and Y. Li, "Lightweight Authentication Protocol for Unmanned Aerial Vehicles Using Physical Unclonable Function and Chaotic System," in *Proc. IEEE LANMAN*, 2020, pp. 1–6.