Defending against Flooding Attacks in the Internet of Drones Environment

Cong Pu and Pingping Zhu
Dept. of CSEE, Marshall University, Huntington, WV 25755, USA. Email: {puc, zhup}@marshall.edu

***Abstract*—Even though drones are still in the infancy period in terms of widespread adoption and use, they have already pierced through solid conventional barriers in various domains of industry. As the usage of drones is becoming commonplace, a next generation aerial communication paradigm, Internet of Drones (IoD), has been proposed to further explore drone technology in a broad scope. IoD relies on the mobility of drones and intermittent drone-to-drone (D2D) and drone-to-ground station (D2I) communications for information sharing and exchange. Because of high mobility and resource constraints, IoD is defenseless to flooding attacks where an adversary sends an excessive amount of packets (original or replica) to legitimate drones with the intention of draining the limited IoD resources (i.e., communication bandwidth and drones' storage space). In this paper, we propose a lightweight distributed detection scheme, hereafter referred to as *Lids*, to defend against flooding attacks in the IoD environment. The basic idea of *Lids* is that each drone counts the number of packets that it has sent within a predefined time interval and shares the self-counting report with other drones during contacts. The receiving drones store the self-counting reports while flying and send them to nearby ground station which will check the consistency of self-counting reports to detect flooding attacks. For performance evaluation, we implement *Lids* and its counterparts (i.e., DAFA and LFADefender) in OMNeT++ network simulator and conduct extensive experiments in terms of detection ratio, miss detection ratio, detection latency, as well as energy consumption. Our experimental results indicate the superior performance of *Lids* to defend against flooding attacks in the IoD environment.**

***Index Terms*—Drone, Internet of Drones (IoD), Flooding Attacks, Lightweight Distributed Detection**

I. Introduction

From a military weapon, to an entertainment tool, to a promising machinery which can revolutionize commercial and civilian industries, the potential of drones is being constantly exploited in the 21st century, and future opportunities in various industry domains are boundless. According to ''Global Drone Market Report 2020-2025'' from Drone Industry Insights [1], the international drone market is estimated to be around $45 billion by 2024, which will triple the 2018 level. The demand for drones by various units around the world is high because drones can be flexibly deployed for a wide range of applications. An eloquent example is that the United Nations Children's Fund (UNICEF) has provided guidance on how to use drones to pick up and delivery lab sample, disinfect contaminated places, and monitor public space during the COVID-19 pandemic [2]. Taking advantage of other advanced technology (i.e., 5G mobile communication, artificial intelligence (AI), virtual reality (VR)), we predict that drone technology will reshape the way we interact with the rest of the world in the foreseeable future.

In order to get more out of drones, a novel aerial communication architecture, Internet of Drones (IoD) [3], has gotten into the spotlight. IoD is composed of mobile drones and stationary ground stations, and supports drone-to-drone (D2D) and drone-to-ground station (D2I) communications. Here, the ground station acts as access point providing real-time communication with drones which are within its controlled airspace. To facilitate mutual exchange of information, a drone can communicate with other drones within its communication range via D2D communication, or interact with nearby ground station through D2I communication. Due to the lack of persistent connectivity between drone and drone and between drone and ground station, the store-carry-and-forward strategy [4] is regarded to be the most promising candidate for delivering data in the IoD environment [5]. By following store-carry-and-forward strategy, a drone stores the received packets in the storage, carries them while flying around, and forwards them to next-hop drone or destination (i.e., ground station).

As a result of high mobility and resource constraints, IoD is vulnerable to flooding attacks where an adversary sends an excessive amount of packets (original or replica) to legitimate drones with the intention of draining the limited IoD resources such as communication bandwidth and drones' storage space. First, the link expiration time between drone and drone or between drone and ground station is relatively short because of the high mobility of drones. Thus, a mass of attack packets can waste precious contact/communication time. Second, the storage capacity of drones is limited. Buffering attack packets prevents legitimate drones from receiving and storing genuine packets. Last but not least, receiving and sending a large amount of attack packets consume non-negligible energy power. For most of battery-powered drones, it is like committing slow suicide. In light of these, it is imperative to investigate flooding attacks and design state-of-the-art countermeasures in the IoD environment.

Flooding attacks are an old research topic, and have been investigated in diverse environments, e.g., traditional computer network [6], named data networking [7], wireless ad hoc network [8], and vehicular ad hoc networks [9]. However, no/low mobility is considered in the abovementioned environments, which makes the proposed countermeasures not able to effectively defend against flooding attacks in the IoD environment. A great deal of effort has been put in by researchers to design a variety of authentication protocols and protect communica-

tions in the IoD environment [10]. Nonetheless, authentication protocol is a type of cryptographic protocol that specifically protects data transmissions from unauthorized entities, and fails to secure the system from the insider attackers who have complete access to cryptographic credentials. To the best of our knowledge, there is no available work concentrating on flooding attacks and their countermeasures in the IoD environment, our work will fill this research gap.

In this paper, we propose a countermeasure and associated techniques to effectively defend against flooding attacks in the IoD environment. In addition, we conduct extensive experiments to evaluate the performance of countermeasure. Our key contribution is summarized as follows:

1) We propose a lightweight distributed detection scheme (*Lids*) to defend against flooding attacks in the IoD environment. In *Lids*, each drone counts the number of packets that it has sent within a predefined time interval and shares the self-counting report with other drones during contacts. The receiving drones store the self-counting reports while flying and send them to nearby ground station which will check the consistency of self-counting reports to detect flooding attacks.
2) We implement *Lids* in OMNeT++ network simulator [11] and conduct extensive experiments in terms of detection ratio, miss detection ratio, detection latency, as well as energy consumption. We also revisit two counterparts such as DAFA [12] and LFADefender [13], and implement them in the network framework for performance comparison and analysis.

The remaining of the paper is organized as follows. In Section II, we present and analyze the existing work. *Lids* and its associated techniques are introduced in Section III. Simulation results and their analysis are provided in Section IV. Finally, we conclude the paper in Section V.

II. Related Work

Over the last several years, security attacks and their defense mechanisms have been studied in flying ad hoc networks. The author in [14] proposes a multi-path packet forwarding scheme to defend against jamming attack in flying ad hoc networks, where three network metrics such as link quality, traffic load, and spatial distance are adopted to select multiple paths between source and destination. Through multi-path routing, network performance (i.e., packet delivery ratio) in complex cyberthreat environment can be improved. However, the computational overhead is also increased accordingly due to the frequent metrics calculation. In [15], a clustering scheme based on fuzzy logic model is proposed to protect flying ad hoc networks from bad mouth and packet dropping attacks. The behaviors of drones are first evaluated and then converted into trust, which is used to distinguish legitimate and malicious drones. The proposed approach is able to achieve a lower packet drop ratio according to performance evaluation. However, the major drawback is the non-negligible energy consumption from cluster maintenance and fuzzy logic system.

As the first line of defense, authentication naturally becomes the center of researchers' attention. In [16], the authors propose an authentication scheme to protect data transmission between ground station and drone based on physical unclonable function and Duffing map. Through random shuffling with secret challenge-response pair, drone and ground station can successfully authenticate each other and establish a secure session key. The authors in [17] propose a blockchain-based data management framework for the IoD environment, where drones and ground station can establish secure communications through access control mechanism and secure session key. Additionally, they also design a consensus algorithm for the competition of adding the blocks in the private blockchain. However, according to the analysis provided by [18], the abovementioned blockchain-based data management framework has several serious vulnerabilities that make the IoD system very fragile when suffering from impersonation attack, man-in-the-middle attack, and replay attack. In [19], a signal strength based secret key generation scheme is developed for flying ad hoc networks, where the received signal strength (RSS) measured at the drone and ground station is used to establish the secret key. However, how to resolve the issue of secret key bit mismatch is a non-trivial problem.

The authors in [20] investigate route request (RREQ) flooding attack and propose a countermeasure in wireless ad hoc networks. To be specific, Bayesian Inference is adopted to model and detect persistent RREQ flooding attack, while Dempster-Shafer evidence theory is developed to defend against flooding attack with various RREQ rate. However, the proposed countermeasure is designed based on an implicit assumption that there is no mobility in the network. Thus, it is not applicable in the IoD environment. In [21], the authors implement an intrusion detection system (IDS) with deep neural network technique to combat data flooding attack in mobile ad hoc networks. The IDS can deliver acceptable detection accuracy. Unfortunately, running deep neural network on resource-constrained mobile devices consumes a significant amount of energy power. The authors in [22] propose a flooding attack defense scheme (Sentinel) in software defined vehicular network. Sentinel is composed of two phases: detection and mitigation. During the detection phase, the packet traffic of each vehicle is first monitored using time series analysis. Then, the statistics and a set of traffic flow rules are applied to detect flooding attack. In the mitigation phase, a flow tree is generated to localize the vehicles who generate malicious traffic. However, Sentinel requires dense placement of road side units for traffic monitoring, which significantly increases deployment costs and operational expenditures.

One may observe that two important issues should be addressed in the design of countermeasure against flooding attacks in the IoD environment: (i) intermittent connectivity in the IoD; and (ii) integration with off-the-shelf routing protocols. First, taking advantage of ''store-carry-and-forward'' strategy, our scheme *Lids* can smoothly solve knotty intermittent connectivity problem. Second, in order to be easily integrated with existing routing protocols, *Lids* is intentionally

designed as a network layer add-on module. Note that, in-depth analysis of flooding attacks and their corresponding countermeasures are not currently available in the IoD environment, and the proposed work will fill this research gap.

III. The Proposed Countermeasure

A. System and Adversary Model

We consider a generic scenario (i.e., search and rescue) in the IoD environment, where a set of drones is deployed to collect interested data and collaboratively transfer them to nearby ground station for further processing. When a drone detects an event, it generates data packets and sends them toward nearby ground station via multi-hop drone relaying. Here, each data packet can be uniquely identified using source drone ID and packet sequence number. Since drones have high mobility, an end-to-end forwarding path does not always exist. As a result, the store-carry-and-forward strategy [5] is adopted in the system, where a drone stores the received packets in the storage, carries them while flying around, and forwards them to next-hop drone or destination (i.e., ground station). However, due to cost constraints, each drone has limited storage space on board. When the storage space is full, a drone cannot receive and store any new packets. In order to purge stale packets in the storage space, each packet is associated with a lifetime. When the lifetime expires, the packet is removed from the storage space. We assume that a public-key cryptography, e.g., lightweight identity-based encryption [23], is being utilized in the IoD environment. We also assume loose time synchronization among drones and ground station.

Since drones are flying in wide-open airspace, there are chances for adversary to physically capture and compromise a legitimate drone to behave maliciously. The primary goal of malicious drones is to flood a large number of original or replica packets to legitimate drones in order to drain the limited IoD resources such as communication bandwidth and drones' storage space. Compromised drones might have obtained cryptographic credentials. Thus, in this paper, we mainly focus on flooding attacks which cannot be detected by cryptographic primitives.

B. Lids: Lightweight Distributed Detection Scheme

The basic idea of *Lids* is that the drone itself counts the number of sent packets within a time period and shares the self-counting report with other drones during contacts. The receiving drones finally deliver the self-counting reports to the ground station, where the self-counting reports will be checked to detect flooding attacks. More details about *Lids* are provided in the following.

First, when drone ID_a joins the IoD environment, it registers at the certificate authority (CA). In this paper, the CA is regarded as a trusted party and not colluding with any adversary. During the registration phase, the CA and drone ID_a negotiate an agreement on the packet send rate RT_a^{pkt}, which indicates the number of packets that drone ID_a is eligible to send within a pre-defined time period T^{ω}. If more than RT_a^{pkt} packets were sent by drone ID_a within T^{ω}, it is considered as an adversary who launched flooding attacks. After that, the CA issues a digital certificate $CERT_a$ to drone ID_a. The digital certificate $CERT_a$ contains drone's identity ID_a, drone's public key PU_a, drone's packet send rate RT_a^{pkt}, and CA's digital signature. The CA also generates a private key PR_a, which is provided to drone ID_a via an independent and secure channel.

Algorithm 1: Drone Detects Flooding Attack

Input: $RPT_m^{T_x}$, $CERT_m$

Function DroneDetect($RPT_m^{T_x}$, $CERT_m$):
- /* verify message authentication code */
- $MAC'_{T_x} = H(CNT^{pkt} \mid T_x \mid CERT_m \mid SIG_m)$;
- **if** $MAC'_{T_x} == MAC_{T_x}$ **then**
  - /* verify digital signature */
  - /* $D(\cdot)$ is a digital signature verification algorithm */
  - $SIG'_m = D(ID_m \mid CERT_m \mid PU_m)$;
  - **if** $SIG'_m == SIG_m$ **then**
    - /* compare the count of sent packets with the packet send rate */
    - **if** $RPT_m^{T_x}.CNT^{pkt} > CERT_m.RT_m^{pkt}$ **then**
      - /* drone ID_m sends more packets than the packet send rate */
      - discard received packets;
    - **else**
      - store received packet;
    - **end**
  - **else**
    - /* digital signature verification failed */
    - discard received packets;
  - **end**
- **else**
  - /* message authentication code verification failed */
  - discard received packets;
- **end**

Second, when drone ID_a comes in contact with another drone ID_b at the time T_x, drone ID_a first sends previously scheduled packets to drone ID_b. Then, drone ID_a sums up the number of sent packets (including the ongoing contact) CNT^{pkt} since the beginning of current time interval T_i^{ω}, creates the self-counting report $RPT_a^{T_x}$, and shares $RPT_a^{T_x}$ with drone ID_b. Here, T_i^{ω} is the ith time interval and T_x is the contact time between drone ID_a and drone ID_b. If drone ID_a does not have any packet to send, it will not create and share the self-counting report with drone ID_b. The self-counting report $RPT_a^{T_x}$ contains the count of sent packets CNT^{pkt}, the contact time T_x, drone ID_a's digital certificate $CERT_a$, drone ID_a's digital signature SIG_a, and a message authentication code MAC_{T_x}. Since $CERT_a$ already includes ID_a, the identity of drone is omitted in $RPT_a^{T_x}$. $RPT_a^{T_x}$ can be represented as follows:

$$RPT_a^{T_x} = \{CNT^{pkt}, T_x, CERT_a, SIG_a, MAC_{T_x}\}. \quad (1)$$

Here, SIG_a and MAC_{T_x} is respectively calculated as

$$SIG_a = E(ID_a|CERT_a|PR_a), \quad (2)$$

and

$$MAC_{T_x} = H(CNT^{pkt}|T_x|CERT_a|SIG_a), \quad (3)$$

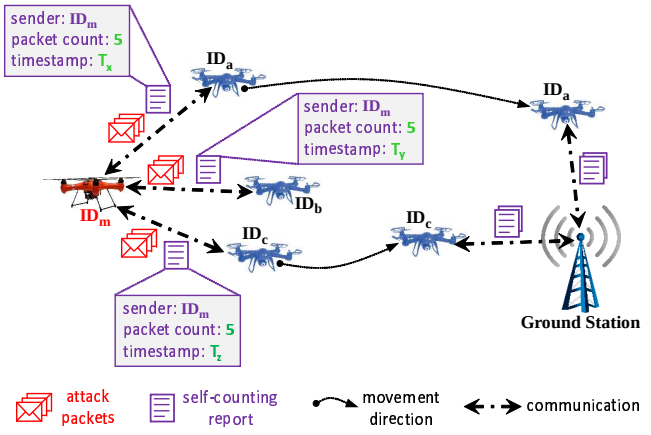


Fig. 1. A system snapshot, where malicious drone ID_m floods packets to legitimate drone ID_a, ID_b, and ID_c. Here, $T_x < T_y < T_z$.

where ''|'' represents the concatenation operation, $H(\cdot)$ is a secure one-way hash function, and $E(\cdot)$ is a digital signature generation algorithm.

After receiving the report $RPT_a^{T_x}$, drone ID_b first verifies whether there is a communication error by comparing the received MAC_{T_x} with its calculated MAC'_{T_x}. If $MAC_{T_x} \neq MAC'_{T_x}$, drone ID_b discards received packets and self-counting report $RPT_a^{T_x}$. Otherwise, drone ID_b proceeds to verify the digital signature SIG_a through digital signature verification algorithm $D(\cdot)$. If the verification succeeds, drone ID_b retrieves the packet send rate RT_a^{pkt} from the digital certificate $CERT_a$, and compares it with the count of sent packets CNT^{pkt}. If $CNT^{pkt} > RT_a^{pkt}$, which indicates drone ID_a sends more packets than its packet send rate, drone ID_b discards all received packets. Drone ID_b does not discard the self-counting report $RPT_a^{T_x}$ since the report will be delivered to the ground station for the detection of flooding attacks. If $CNT^{pkt} \leq RT_a^{pkt}$, drone ID_b carries the received packets and delivers them to next-hop drone or destination (i.e., ground station). The algorithm of drone detecting flooding attack is described in Algorithm 1.

Third, when drone ID_b reaches the ground station, it submits all received self-counting reports. The ground station will compare the newly received reports with the already obtained reports to identify whether a drone issued multiple reports with inconsistent information and then detect flooding attacks. The rationale behind the detection is that if a malicious drone sends more packets than its packet send rate within a time interval, in order to avoid detection, it needs to disloyally report a false packet count which is smaller than or equal to the actual packet count (or its packet send rate). However, the false packet count must have been reported before by the malicious drone, or the false packet count is smaller than or equal to a packet count which was reported in an earlier self-counting report. As a result, the ground station can easily identify inconsistent information existing in the self-counting reports and detect flooding attacks.

For example, as shown in Fig. 1, we consider a snapshot of system, where malicious drone ID_m is flooding packets to legitimate drone ID_a, ID_b, and ID_c. Here, we assume that

Algorithm 2: Ground Station Detects Flooding Attack

Input: RPT

```
/* RPT is a set of received self-counting reports */
Function GStationDetect(RPT):
    /* evaluate each issuer of self-counting report */
    /* G_isu is a set of self-counting report issuers */
    for i ∈ |G_isu| do
        /* check each self-counting report from drone ID_i; */
        /* G^i_rpt is a set of self-counting reports issued by drone ID_i */
        for j ∈ |G^i_rpt| do
            /* check whether drone ID_i sends more packets than the packet send rate */
            if RPT_i^{T_j}.CNT^pkt > CERT_i.RT_i^pkt then
                /* detect flooding attack */
                ψ_i = ψ_i + 1;
            end
        end
        /* check whether drone ID_i issues multiple self-counting reports with inconsistent information */
        for j ∈ |G^i_rpt| do
            for k ∈ [1, j) do
                if RPT_i^{T_k}.CNT^pkt >= RPT_i^{T_j}.CNT^pkt then
                    /* detect flooding attack */
                    ψ_i = ψ_i + 1;
                end
            end
        end
        /* isolate drone ID_i from the system */
        if ψ_i >= TH_iso then
            broadcast Alarm packet;
        end
    end
```

malicious drone ID_m's packet send rate RT_m^{pkt} = 5, and the timestamp $T_x < T_y < T_z$. First, malicious drone ID_m sends five packets to drone ID_a during their contact. Accordingly, malicious drone ID_m creates the self-counting report $RPT_m^{T_x}$ and shares it with drone ID_a. $RPT_m^{T_x}$ contains the count of sent packets CNT^{pkt} = 5, which reaches the limit of ID_m's packet send rate. Later, malicious drone ID_m communicates with drone ID_b and sends another five packets. However, this time malicious drone ID_m needs to report a false count of sent packets, e.g., CNT^{pkt} = 5, in the self-counting report $RPT_m^{T_y}$, in order to avoid the detection by drone ID_b. This is because if malicious drone ID_m loyally reported the actual count of sent packets CNT^{pkt} = 10, drone ID_b can easily detect the inconsistent information in the self-counting report $RPT_m^{T_y}$ and digital certificate $CERT_m$, and drops all received packets without forwarding further. Thus, malicious drone ID_m has to falsely report the count of sent packets CNT^{pkt} = 5 in the self-counting report $RPT_m^{T_y}$. However, the false reporting can be detected later by the ground station. When malicious drone ID_m meets drone ID_c, it sends five packets and falsely reports the count of sent packets, e.g., CNT^{pkt} = 5, in the self-counting report $RPT_m^{T_z}$ again. When drone

TABLE I
SIMULATION PARAMETERS

Parameter	Value
Network area	150×150 m^2
Number of legitimate drones	35
Number of malicious drones	5
Number of ground stations	3
Moving speed	15 meter/sec
Mobility model	Random waypoint
Communication range of drone	12.59 meters
Communication range of ground station	50 meters
Radio data rate	3 Mbit/sec
Packet size	127 bytes
Attack packet rate	1.0 - 4.0 pkt/sec
Simulation time	10,000 seconds

ID_a and ID_c reach the ground station, they submit their received self-counting reports including $RPT_m^{T_x}$ and $RPT_m^{T_z}$. The ground station checks $RPT_m^{T_x}$ and $RPT_m^{T_z}$ and realizes that they piggyback different timestamps (T_x and T_z) but the same count of sent packets CNT^{pkt} = 5. As a result, malicious drone ID_m is suspected for launching flooding attacks. When the number of detection ψ_m reaches a threshold value TH_{iso}, the ground station broadcasts an *Alarm* packet to all drones in the system to prevent them from receiving any packets from malicious drone ID_m. The algorithm of ground station detecting flooding attack is described in Algorithm 2.

IV. PERFORMANCE EVALUATION

To evaluate the performance of *Lids*, we develop a simulation framework in OMNeT++ [11], where 40 drones including five adversaries and three ground stations are deployed in a 150×150 network area. According to log-distance path loss model, the signal strength threshold is set to -81dBm so that each drone has a wireless transmission of 12.59 meters. The wireless coverage range of ground station is 50 meters. The random waypoint mobility model is adopted in the framework, where each drone moves with a constant speed of 15 meter/sec. In addition, the packet size is 127 bytes and the data rate is set to 3 Mbit/sec. The storage size of each drone is 100 packets and the packet send rate RT^{pkt} = 10. The radio mode is assumed to be ideal, i.e., transmissions within the communication range from a transmitter are perfectly received, and outside this range not received at all. The total length of simulation is 10,000 seconds. We measure the performance in terms of detection ratio, miss detection ratio, detection latency, as well as energy consumption. Moreover, the experiment is repeated 5 times with different simulation seed to obtain the steady performance result of each metrics. For performance comparison, we also implement two counterparts, DAFA [12] and LFADefender [13]. In DAFA, if a drone's average packet transmission rate is larger than the threshold value, such a drone is suspected as an adversary. In LFADefender, the link performance metrics such as packet loss rate, round-trip time, and available bandwidth are being monitored to determine whether flooding attacks are happening. The simulation parameters are summarized in Table I.

We measure the performance of detection ratio with varying attack packet rate in Fig. 2(a). Here, a larger attack packet

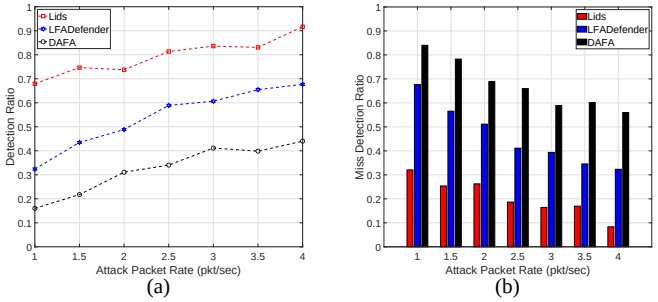
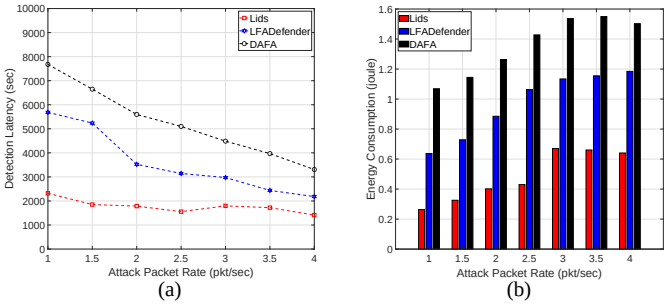


Fig. 2. The performance of detection ratio and miss detection ratio against attack packet rate.

rate indicates more attack packets will be issued per second by an adversary. Overall, the detection ratio of *Lids*, DAFA and LFADefender increase as the attack packet rate increases. However, the highest detection ratio is obtained by *Lids*. When the attack packet rate reaches 4 pkt/sec, the detection ratio of *Lids* is maintained above 90%. In *Lids*, each drone carries the self-counting reports issued by adversaries and directly submit them to the ground station who can easily detect flooding attacks. As a result, more attempts of flooding attacks can be detected by the ground station, and a larger detection ratio can be obtained. The lowest detection ratio is observed by DAFA. This is because DAFA adopts a threshold value, compares it with the packet rate, and determines whether flooding attacks exist or not. However, the number of received attack packets might be lower than the pre-defined threshold value. Thus, the on-going flooding attacks cannot be detected and a lower detection ratio is measured. LFADefender delivers a higher detection ratio compared to DAFA. Since LFADefender adopts three different link metrics, i.e., packet loss rate, round-trip time, and available bandwidth, it can more accurately detect flooding attacks. The miss detection ratio is measured by changing attack packet rate in Fig. 2(b). Compared to DAFA and LFADefender, *Lids* can achieve the lowest miss detection ratio. Instead of comparing the number of received attack packets with a threshold value, the ground station in *Lids* evaluates all received self-counting reports to detect inconsistent information due to flooding attacks, thus the number of missed detection can be significantly reduced. LFADefender observes a lower miss detection ratio compared to that of DAFA. Since more flooding attacks can be detected by LFADefender, a lower miss detection ratio is obtained by LFADefender.

Fig. 3(a) shows the detection latency of *Lids*, DAFA and LFADefender with varying attack packet rate. In this experiment, the detection latency means how soon the adversary can be isolated from the system via broadcasting an *Alarm* packet. At first glance, as the attack packet rate increases from 1 to 4 pkt/sec, the detection latency of three schemes decline accordingly. However, *Lids* still outperforms DAFA and LFADefender. Since more flooding attacks can be detected by *Lids*, the number of attack detection can quickly reach the threshold value. As a result, the adversary can be quickly removed from the system. As the attack packet rate increases, the detection latency of DAFA and LFADefender quickly decrease. This is because a larger attack packet rate can make



(a) (b)

Fig. 3. The performance of detection latency and energy consumption against attack packet rate.

flooding attack more easily to be detected. Thus, the system can broadcast *Alarm* packet earlier to isolate the adversary from the system. In Fig. 3(b), we show the energy consumption of *Lids*, DAFA and LFADefender due to flooding attacks. Here, the energy consumption of flooding attacks is measured based on the number of received and forwarded attack packets [24]. Again, the best performance belongs to *Lids*. The highest energy consumption is observed by DAFA, while LFADefender delivers a medium performance. In this experiment, if the adversary was isolated from the system, legitimate drones will not accept and forward any attack packets from the adversary. As a result, a lower energy consumption will be obtained. Following this idea, *Lids* can remove the adversary at the earliest possible time, thus, the lowest energy consumption is obtained by *Lids*. This is because legitimate drones in *Lids* do not receive and forward any attack packets after the adversary is removed from the system.

V. Conclusion

In this paper, we proposed a lightweight distributed detection scheme (*Lids*) to detect and mitigate flooding attacks in the IoD environment. The basic idea is that each drone counts the number of packets that it has sent within a predefined time interval and shares the self-counting report with other drones during contacts. The receiving drones store the self-counting reports while flying and send them to nearby ground station which will check the consistency of self-counting reports to detect flooding attacks. This is the first work that investigated flooding attacks and then proposed a corresponding countermeasure in the IoD environment. Thus, an existing research gap has been filled. Through extensive experimental study, we found that *Lids* can improve detection ratio as well as reduce miss detection ratio, detection latency, and energy consumption. In *Lids*, one constraint needs to be further investigated. For example, when drones frequently meet and exchange the packets, a large number of self-counting reports will be generated in the network. How to efficiently and effectively share those self-counting reports among drones and ground stations becomes a challenging problem. As a future work, we plan to propose a data reduction strategy and develop a read-world testbed to explore the full potential of *Lids*.

References

[1] *Drone Industry Insight*, Last accessed: August 31, 2021, https://droneii.com/.

[2] *How Drones Can Be Used to combat COVID-19*, Last accessed: August 31, 2021, https://www.unicef.org/supply/documents/how-drones-can-be-used-combat-covid-19.

[3] 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.

[4] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay-Tolerant Networking Architecture," *IETF RFC 4838*, 2007.

[5] C. Pu and L. Carpenter, "To Route or To Ferry: A Hybrid Packet Forwarding Algorithm in Flying Ad Hoc Networks," in *Proc. IEEE NCA*, 2019, pp. 1–8.

[6] A. Aydeger, M. Manshaei, M. Rahman, and K. Akkaya, "Strategic Defense against Stealthy Link Flooding Attacks: A Signaling Game Approach," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 1, pp. 751–764, 2021.

[7] C. Pu, N. Payne, and J. Brown, "Self-Adjusting Share-Based Countermeasure to Interest Flooding Attack in Named Data Networking," in *Proc. IEEE CPSCom*, 2019, pp. 142–147.

[8] N. Nishanth and A. Mujeeb, "Modeling and Detection of Flooding-Based Denial-of-Service Attack in Wireless Ad Hoc Network Using Bayesian Inference," *IEEE Systems Journal*, vol. 15, no. 1, pp. 17–26, 2021.

[9] P. Singh, A. Agarwal, G. Nakum, D. Rawat, and S. Nandi, "MPFSLP: Masqueraded Probabilistic Flooding for Source-Location Privacy in VANETs," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 10, pp. 11 383–11 393, 2020.

[10] M. Yahuza, M. Idris, I. Ahmedy, A. Wahab, T. Nandy, N. Noor, and A. Bala, "Internet of Drones Security and Privacy Issues: Taxonomy and Open Challenges," *IEEE Access*, vol. 9, pp. 57 243–57 270, 2021.

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

[12] P. Mohammadi and A. Ghaffari, "Defending Against Flooding Attacks in Mobile Ad-Hoc Networks Based on Statistical Analysis," *Wireless Personal Communications*, vol. 106, no. 2, pp. 365–376, 2019.

[13] J. Wang, R. Wen, J. Li, F. Yan, B. Zhao, and F. Yu, "Detecting and Mitigating Target Link-Flooding Attacks Using SDN," *IEEE Trans. Dependable Secure Comput.*, vol. 16, no. 6, pp. 944–956, 2019.

[14] C. Pu, "Jamming-Resilient Multipath Routing Protocol for Flying Ad Hoc Networks," *IEEE Access*, vol. 6, pp. 68 472–68 486, 2018.

[15] K. Singh and A. Verma, "TBCS: A Trust Based Clustering Scheme for Secure Communication in Flying Ad-Hoc Networks," *Wireless Personal Communications*, vol. 114, pp. 3173–3196, 2020.

[16] 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.

[17] B. Bera, S. Saha, A. Das, N. Kumar, P. Lorenz, and M. Alazab, "Blockchain-Envisioned Secure Data Delivery and Collection Scheme for 5G-Based IoT-Enabled Internet of Drones Environment," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 9097–9111, 2020.

[18] S. Chaudhry, K. Yahya, M. Karuppiah, R. Kharel, A. Bashir, and Y. Zikria, "GCACS-IoD: A certificate based generic access control scheme for Internet of drones," *Computer Networks*, vol. 19, p. 107999, 2021.

[19] K. Li, N. Lu, J. Zheng, P. Zhang, W. Ni, and E. Tovar, "BloothAir: A Secure Aerial Relay System Using Bluetooth Connected Autonomous Drones," *ACM Transactions on Cyber-Physical Systems*, vol. 5, no. 3, pp. 1–22, 2021.

[20] N. Nishanth and A. Mujeeb, "Modeling and Detection of Flooding based Denial of Service Attacks in Wireless Ad Hoc Networks using Uncertain Reasoning," *IEEE Transactions on Cognitive Communications and Networking (Early Access)*, pp. 1–1, 2021.

[21] O. Sbai and M. E. boukhari, "Data Flooding Intrusion Detection System for MANETs Using Deep Learning Approach," in *Proc. ACM SITA*, 2020, pp. 1–5.

[22] G. de Biasi, L. Vieira, and A. Loureiro, "Sentinel: Defense Mechanism against DDoS Flooding Attack in Software Defined Vehicular Network," in *Proc. IEEE ICC*, 2018, pp. 1–6.

[23] C. Lin, D. He, N. Kumar, K. Choo, A. Vinel, and X. Huang, "Security and Privacy for the Internet of Drones: Challenges and Solutions," *IEEE Communications Magazine*, vol. 56, no. 1, pp. 64–69, 2018.

[24] 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.