IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Distributed Misbehavior Detection in UAV Flocks

MOHAMED ANIS AGUIDA[1], (Member, IEEE), SERGIO A. SALINAS MONROY [1], (MEMBER, IEEE), AND MONOWAR HASAN[2], (Member, IEEE)

[1]School of Computing, Wichita State University, Wichita, KS, USA (e-mail: [mohamedanis.aguida,sergio.salinasmonroy]@wichita.edu)
[2]School of Electrical Engineering & Computer Science, Washington State University, Pullman, WA, USA (e-mail: [monowar.hasan@wsu.edu)

Corresponding author: Sergio A. Salinas Monroy (e-mail: sergio.salinasmonroy@wichita.edu).

ABSTRACT Unmanned aerial vehicles have become increasingly popular in many applications such as remote surveillance, reconnaissance, and precision agriculture. Often multiple UAVs form a swarm and perform their operations in a distributed, coordinated fashion. A rogue UAV in the flock can negatively disrupt the expected behavior and may jeopardize the objective of the mission, leading other UAVs to make incorrect decisions or even crash. This paper introduces GRIFFIN, a *distributed* and *lightweight* misbehavior detection framework for UAV flocks. GRIFFIN relies on readily available GPS packets (i.e., GPS coordinates) and signal characteristics (e.g., RSSI measurements) and detects malicious UAVs by employing a "majority voting" protocol. We show that GRIFFIN requires only four honest nodes for correct operations. We implement and evaluate GRIFFIN on *(a)* a realistic UAV simulator (ArduSim) and *(b)* a Raspberry Pi+Navio-based drone testbed for runtime overhead. We find that GRIFFIN outputs a high detection rate with low false negatives against multiple types of adversaries, as long as the swarm remains predominantly benign. Our implementation on the real UAV testbed shows that the runtime overhead of GRIFFIN is minimal (i.e., it requires less than $\mathbf{1}$ MB of memory and consumes less than $\mathbf{1\%}$ of CPU) and completes its computation within $\mathbf{2.5}$ ms.

INDEX TERMS Unmanned aerial vehicles; UAV swarms; distributed misbehavior detection; rogue UAV detection; swarm security

I. INTRODUCTION

UNMANNED aerial vehicles (UAVs) are becoming increasingly popular in many military, civilian, and commercial applications, such as remote surveillance, border patrol, battlefield monitoring, rescue, disaster management, and precision agriculture, to name a few [1]–[4]. Often the UAVs are required to coordinate in a *distributed*, *cooperative* manner to accomplish the desired mission objectives. For example, a cooperative fleet of UAVs can increase the coverage of reconnaissance missions. A swarm of UAVs can often carry out tasks that cannot be satisfyingly performed with a single UAV. Examples of such cases include: *(a)* when simultaneous measurements are necessary or *(b)* when the mission objective is too complex to be performed by a single UAV with its limited resources (e.g., memory, storage, battery) [5]. In such cases, cooperative coordination among UAVs allows load balancing and computation offloading.

While cooperative communication aids in carrying out mission objectives, a malicious, misbehaving UAV can disrupt the regular operation of the flock, leading to deviating from the mission or even crashing the other UAVs. Consider a reconnaissance mission where UAVs in a swarm collaborate to perform an exploration of a specific area. While in the mission, the UAVs broadcast their positions. The position information can be used by other UAVs to *(a)* determine if an area has already been explored by a specific UAV and *(b)* avoid a collision. A malicious UAV within the fleet can share a wrong position which may jeopardize the objective of the mission (e.g., by sending a wrong position, the mission coordinator may mark an area as explored where it is not) or may deviate from the mission (e.g., force other UAVs operating through a no-fly-zone). Further, such crafted, false location information may mislead about the "true position" of UAVs — leading them to collide and, eventually, crash. Hence, there is a need for designing techniques that can effectively detect misbehaving nodes in a UAV flock.

Prior work has studied UAV swarm security from different angles, such as defense swarms against malicious

UAV intrusions [6], secure multi-UAV communication patterns [7], and anomaly-based intrusion detection for UAV swarms [8]. However, detecting false position reports in a *lightweight* and *fully distributed* manner remains a challenging problem [9]. Our work addresses this gap by verifying GPS-reported positions against RSSI-derived distances and by using jury-based geometric verification with distributed majority voting, without requiring additional sensors, cameras, or centralized infrastructure.

This paper presents a framework (named GRIFFIN) to detect incorrect location information in UAV swarms. GRIFFIN works in a "distributed" manner — which makes it resilient against a single point of failure (i.e., adversarial manipulations of the checker node). GRIFFIN[1] relies on readily available GPS packets (i.e., GPS coordinates) and physical-layer information (e.g., received signal strength) to detect malicious UAVs in the flock. Such lightweight input requirements allow GRIFFIN to operate without requiring additional peripherals such as sensors or cameras. GRIFFIN is effective in detecting location forging attacks such as GPS jamming/spoofing attacks [10] where an adversary tamper the GPS information and broadcast incorrect positions. The key idea of GRIFFIN is to use *two different sources to verify the position information*: *(a)* the first is to get the GPS location from the UAV (broadcast as part of the mission coordination), and *(b)* the second is to use onboard wireless radios to obtain the received signal strength (RSSI). Upon each packet arrival, GRIFFIN then correlates those two pieces of information and runs a majority voting protocol to determine if a target UAV is legitimate or malicious. This paper makes the following contributions:

- A *distributed* and *lightweight* framework, GRIFFIN, to detect malicious UAV positions in a UAV flock without needing any custom hardware/sensor; and
- A *formal guarantee* on the minimum number of "legitimate" UAVs required in the flock to correctly detect a misbehaving node.

We implement and evaluate GRIFFIN on a realistic UAV simulator (ArduSim [11]). We evaluated the runtime overhead on an off-the-shelf drone platform based on Raspberry Pi [12] and Navio [13]. Our results show that GRIFFIN outputs a high detection rate with low false negatives against multiple types of adversaries, as long as the swarm remains predominantly benign. The runtime overhead of GRIFFIN is minimal — on average, GRIFFIN *(a)* takes 2.5 ms to execute, *(b)* requires less than 1 MB of memory, and *(c)* consumes less than 1% of CPU on our drone testbed.

II. RELATED WORK

Recent work has shown that UAV swarm security involves challenges beyond those found in single-UAV systems, since compromised or misbehaving UAVs can influence cooperative decisions, formation stability, area coverage, and collision avoidance. Wang et al. [14] provide a broad survey of attacks and countermeasures in UAV swarm networks, covering threats against communication, coordination, routing, sensing, and swarm control. Their survey highlights the need for security mechanisms that are lightweight and suitable for cooperative UAV networks. GRIFFIN addresses one concrete problem: it detects authenticated swarm members that falsify their position information during cooperative operation. Brust et al. [6] propose a defense mechanism against a single unauthorized UAV using a coordinated defensive swarm. In contrast, GRIFFIN considers a malicious multi-UAV setting. Raja et al. [7] propose an efficient and secure swarm-pattern communication scheme for multi-UAV systems, focusing on protected communication and swarm connectivity. Such mechanisms are complementary to GRIFFIN: secure communication can protect message exchange and sender identity, whereas GRIFFIN verifies whether the position information reported by an authenticated UAV is physically consistent with RSSI-derived distance and geometric constraints.

DIAT [15] proposes an attestation mechanism that secures the interaction between two devices. Lopez et al. [16] present a security-focused architecture for UAV mesh communications. The authors introduce a new layer (called "mesh security layer") which includes features related to the detection and protection of UAV swarms. Sun et al. [17] focus on the physical security of UAVs where they aim to examine passive and active eavesdropping in UAV wireless communication systems. Other studies also propose techniques to protect the communication channels [18]. While the aforementioned work focuses on analyzing security from the communication perspective, GRIFFIN focuses on identifying the misbehaving UAVs in the swarm.

There is some work on detecting intrusions. Mitchel et al. [19] propose an intrusion detection system (IDS) based on behavior rule specification to quickly assess the "survivability" of the UAV under a malicious attack. The authors consider protecting the swarm from several attacks using designer-specified detection rules. Condomines et al. [20] focus on implementing an IDS based on statistical signatures of transmission control protocol (TCP) and user datagram protocol (UDP). Tan et al. [21] and Cengiz et al. [22] develop a machine-learning–based IDS for UAV networks (deep belief networks optimized by particle swarm optimization, and ANN+GA with feature reduction, respectively), targeting high-dimensional network traces rather than distributed behavioral consistency within a flock. Sedjelmaci et al. [23] propose a method to detect cyber-attacks that target the data integrity and network availability of UAV networks. They aim to minimize false negatives and positives in detecting malicious UAVs in the fleet. Bhattacharya et al. [24] consider a differential game theoretic approach to evade the attack from an aerial jammer on the communication channel. While the

[1]In Greek mythology, "GRIFFIN" is a bird-like character that protects precious materials such as gold. Since our focus here is to protect mission-critical information (e.g., locations or coordinates) from tampering, we name our framework as GRIFFIN.

authors implement a method to enable secure communication between UAVs in the presence of jamming attacks, they do not provide a detection mechanism for jamming attacks. We note that all the aforementioned lines of work is strictly layer-specific (physical, network, or application layer). In contrast, GRIFFIN follows a cross-layer approach where we leverage *(i)* packet metadata (e.g., GPS coordinates), *(ii)* physical layer characteristics (RSSI), *(iii)* the network layer communications for majority voting, and *(iv)* the application layer checks for misbehavior detection. Further, the distributed nature of GRIFFIN makes it resilient against single-node compromises compared to existing centralized solutions [19], [20], [23], [24].

III. MODEL AND ASSUMPTIONS

We now start by presenting the system model (Sec. III-A) before discussing our assumptions on adversarial capabilities (Sec. III-B).

A. SYSTEM MODEL

1) Network and Communication Model

We assume a UAV swarm performing a cooperative mission. We consider a slotted verification model with a processing rate of 1 Hz [25]. We assume n UAVs present in the flock and denote the set of UAVs as $\mathcal{U} = \{u_1, u_2, u_3, \cdots, u_n\}$. At each verification slot, UAVs periodically broadcast timestamped GPS position messages. The position information is carried as a message payload. Each receiving UAV measures the RSSI of the received packet locally and pairs it with the corresponding timestamped position message within the same verification slot. Let $\mathcal{D} = \{d_{ij}(\hat{t}) \mid u_i, u_j \in \mathcal{U}, i \neq j\}$ denote the set of pairwise distances between UAVs at time $\hat{t}$, where $d_{ij}(\hat{t})$ represents the distance between u_i and u_j.[2] We assume that all UAVs are within communication range during the considered mission. Therefore, each UAV can communicate with the other UAVs and with the ground station. GRIFFIN is protocol-agnostic and can operate over different communication technologies, such as WiFi, cellular, or satellite links, as long as the receiver can obtain RSSI measurements and authenticated position messages. We assume that UAV identities are established before deployment and that inter-UAV messages are authenticated using a Public Key Infrastructure (PKI) based signing mechanism. This prevents external impersonation and Sybil attacks, while GRIFFIN focuses on authenticated insider misbehavior, in which compromised UAVs may send false position, distance, or voting information using valid credentials. Figure 1 presents a high-level illustration of the UAV swarm communication setting that triggers GRIFFIN.

2) RSSI Model

The received signal strength indicator (RSSI) describes the relationship between transmitted power, received power, and the distance among nodes. This relationship can be represented as follows [26]: $P_r = P_t \times (1/d)^n$, where P_r is the received power and P_t is the transmitted power of the wireless signal, d is the distance between the transmitter UAV and the receiving UAV (u_t and u_r), and n is the path-loss (transmission) exponent whose value depends on the propagation environment [27]. By taking the base-10 logarithm of the above expression (to more easily handle the exponential relationship), we obtain: $10\log_{10}(P_r) = 10\log_{10}(P_t) - 10n\log_{10}(d)$, where $10\log_{10}(P_r)$ is the received power in dBm and corresponds to the RSSI. Hence, we can express the dependence of RSSI on distance as: $RSSI(d) = A - 10n\log_{10}(d)$, where $A = 10\log_{10}(P_t)$ absorbs the transmit power (and, in more detailed models, additional constant terms such as antenna gains and reference-distance loss).

Let d_{RSSI} denote the distance (based on RSSI) between a pair of UAVs u_i and u_j. Inverting the above relationship yields:

$$d_{\text{RSSI}} = 10^{\frac{A-RSSI}{10n}}. \tag{1}$$

3) GPS Model

To measure the distance between a pair of UAV u_i and u_j, we need to know their locations: the GPS position (including altitude) of u_i and the GPS position of u_j. The UAVs u_i and u_j need to exchange their GPS positions to calculate this distance. We assume that the GPS information exchanged by the UAVs forms a $(\text{latitude}, \text{longitude}, \text{altitude})$ tuple. Since latitude and longitude are angular coordinates, they cannot be directly combined with altitude in a Euclidean distance calculation. Therefore, each GPS tuple is first projected into a metric coordinate system using the Universal Transverse Mercator (UTM) projection [28]. Specifically, a GPS position given as $(\text{latitude}, \text{longitude}, \text{altitude})$ is converted into (E, N, h), where E and N denote the UTM easting and northing in meters, respectively, and h denotes the altitude in meters. For notational simplicity, we denote the projected coordinates of UAV u_i as $u_i(x_i, y_i, z_i)$ and those of UAV u_j as $u_j(x_j, y_j, z_j)$, where x and y correspond to the UTM easting and northing, respectively, and z corresponds

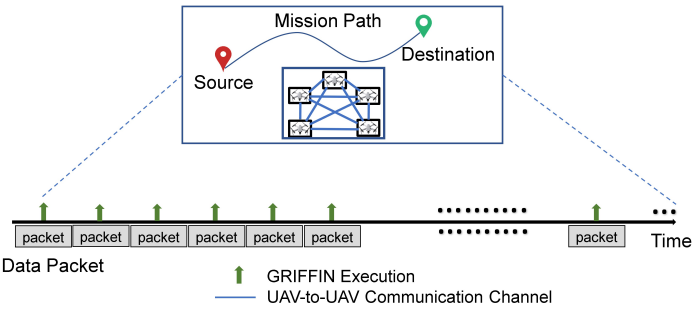


FIGURE 1: High-level illustration of the UAV swarm communication setting and the packet-arrival event that triggers GRIFFIN. The detailed GRIFFIN detection workflow is shown in Fig. 2.

[2]For simplicity of notation, we omit $\hat{t}$ unless it is required.

to altitude. We use Euclidean distance [29] as a distance measure and express the corresponding distance (denoted by d_{GPS}) as follows:

$$d_{GPS} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}. \quad (2)$$

B. THREAT MODEL

We consider an adversary that can compromise a subset of UAVs through software exploitation, supply-chain compromise, physical capture, GPS spoofing/jamming, or social-engineering attacks [10], [24], [30]–[32]. Compromised UAVs remain authenticated members of the swarm and therefore possess valid credentials and protocol knowledge. The adversary can observe inter-UAV communications available to legitimate nodes, falsify its own reported position, selectively provide inconsistent position or distance information to different receivers, provide false measurements when acting as a jury, manipulate its local detection verdicts, collude with other compromised UAVs, and adopt adaptive on–off behavior to evade detection. The adversary aims to remain undetected while disrupting swarm coordination, misleading coverage decisions, causing navigation errors or collisions, protecting malicious UAVs, and falsely accusing benign UAVs. We assume that UAV identities are established prior to mission deployment and that inter-UAV messages are authenticated using PKI. Therefore, the adversary cannot forge messages on behalf of non-compromised UAVs or introduce Sybil identities. Furthermore, we assume that the adversary can manipulate data packets but cannot change system settings, such as radio transmit power to shape the RSSI observed by receivers. Consequently, GRIFFIN focuses on detecting false positions and distance claims rather than physical-layer power-control attacks. Finally, we assume that the adversary cannot compromise the entire swarm or break the underlying cryptographic primitives.

IV. MALICIOUS UAV DETECTION IN GRIFFIN

A. OVERVIEW

We define "misbehavior" as the incorrect reporting of location/position information. Our main goal is to detect malicious UAVs within the flock that report a false position in a distributed manner. As mentioned earlier, we assume that all UAVs are within communication range and broadcast their GPS positions. The framework runs at each time slot. Figure 2 presents a high-level schematic of GRIFFIN. Upon the arrival of each packet [circle ① in the figure]. Each UAV calculates the distance toward all other UAVs using two methods [block ②], viz., *(a)* the RSSI method (Sec. III-A2) and *(b)* the GPS method (Sec. III-A3). The former is based on the signal strength of the received packets, and the latter is based on the received GPS coordinates (included as packet metadata). Next, it compares the two distances [block ③]. If the two distances match within the accepted threshold, GRIFFIN proceeds to the next step [block ④-block ⑤]. Otherwise, the UAV will be marked as malicious (red rectangle). If the GPS distance matches the RSSI distance within the accepted threshold, three juries will be elected to confirm the claimed coordinates (block ④). We follow a process based on sphere radius (i.e., distance between nodes) and Euclidean distance to confirm the position [block 5a-5d]. All UAVs broadcast their detection result. After that, GRIFFIN executes a majority voting algorithm. The key idea is to assign a *"Trust Level"* (TL) to each UAV based on the majority voting results. The TL will be updated by adding -1 if malicious or 1 if legitimate in each round of GRIFFIN. When a malicious UAV exceeds a certain negative Trust Level, we finally mark the UAV as malicious. Once a UAV is tagged as malicious, the ground station may take further actions, such as informing other UAVs to ignore packets from malicious UAVs for the rest of the mission. We discuss this issue further in Sec. VII.

FIGURE 2: High-level workflow of GRIFFIN.

B. DISTANCE COMPARISON MODEL

Once we calculate the distance using two different methods, we need to define the accepted threshold by taking measurement errors into consideration. We define the distance difference as follows:

$$\Delta d = |d_{GPS} - d_{RSSI}|. \quad (3)$$

We consider the distance legitimate when $\Delta d \leq \theta$. That is, the threshold depends on the noise of the RSSI and GPS calculation and on the environment which the UAVs are flying on. The parameter θ captures this error effect.

1) Legitimate GPS Coordination

The distance comparison model alone is not enough to ensure that the target UAV is sending its true position. We explain this problem with a simple illustration. Let us assume that u_t

sends its position to u_r. After u_r performs the calculations using the distance comparison model (Eq. 3), it finds that the distance d_{GPS} is (almost) equal to the distance d_{RSSI}. This information affirms that u_t is in the sphere defined by d_{GPS} as the radius and the position of u_r as the center. In other words, u_t could be anywhere on the sphere surface but with a legitimate distance. We illustrate this scenario in Fig. 3.

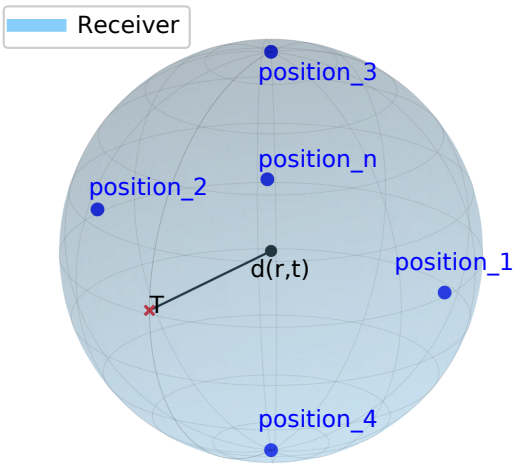


FIGURE 3: Issues with simple distance comparison: the target UAV could be in any position on the sphere, and it is non-trivial to identify its correct position.

To solve this issue, we need techniques to precisely verify the position. For this, we propose a sphere-based calculation method involves one *receiver* (i.e., who determines the legitimacy of the target) and three *juries* (who help the receiver in the decision-making process). Note that with just one jury we retain a circle of candidates in 3D (intersection of two spheres; see Fig. 4), and with two juries we usually still have up to two symmetric points. A *third jury* resolves this ambiguity in 3D and pins down a unique position.

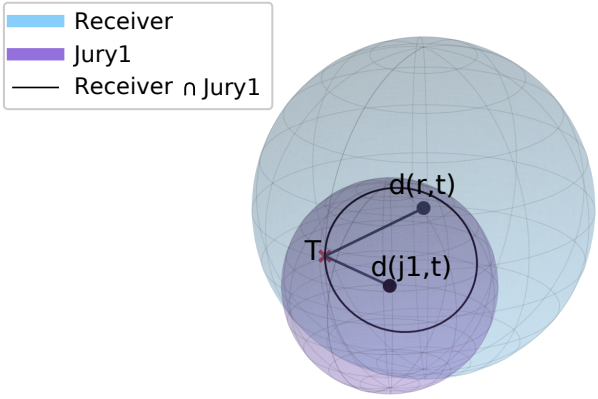


FIGURE 4: Using one jury cannot solve the issues illustrated in Fig. 3.

We denote the receiver as u_r and the three juries as u_{j_1}, u_{j_2}, and u_{j_3}, respectively. The intersection of the four spheres (centered at $u_r, u_{j_1}, u_{j_2}, u_{j_3}$ with their respective distances to u_t as radii) must contain the position of u_t (generically a unique point in 3D). To accomplish this, we check whether the GPS coordinates of the target UAV (u_t) fulfill the sphere equations of u_r, u_{j_1}, u_{j_2}, and u_{j_3} (presented next). In other words, GRIFFIN checks whether u_t is included within the spheres: *(a)* with u_r as the center and d_{tr} as the radius, *(b)* with u_{j_1} as the center and d_{tj_1} as the radius, *(c)* with u_{j_2} as the center and d_{tj_2} as the radius, and *(d)* with u_{j_3} as the center and d_{tj_3} as the radius. If all the conditions are satisfied, we can confirm its position as legitimate. Figure 5 illustrates the overall idea.

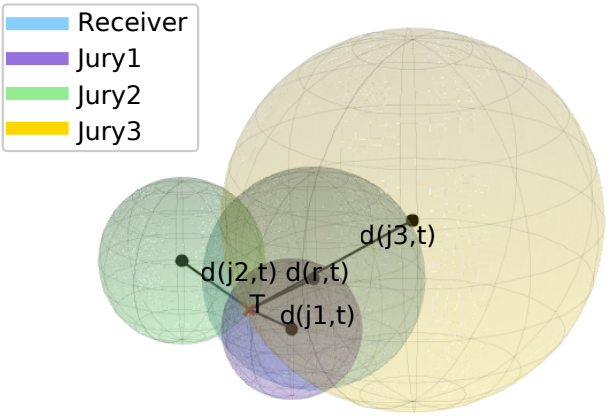


FIGURE 5: Illustration of the malicious UAV detection process using three juries. The receiver checks the information received from all juries and determines the legitimacy of the location broadcast by the target.

The following equation defines a sphere with (x_0, y_0, z_0) as the center and r as the radius (which represents a distance between two UAVs in our case):

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = r^2.$$

This vanilla equation may be affected by noise, say due to the RSSI measurement errors and GPS calculation offsets. To overcome this, we consider an error C to capture the RSSI/GPS noise, and the threshold $\pm\theta$ as the acceptable RSSI/GPS noise. We represent the above equation as follows:

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 - r^2 = \pm C.$$

where $|C| \leq \theta$.

Let us consider first the position of u_r as the center and the distance between u_r and u_t as the radius. If the above equation remains correct, the receiver matches this with the information received from the three juries u_{j_1}, u_{j_2}, and u_{j_3}.

To be specific, all **four** of the following equations must be valid to determine the legitimacy of u_t:

$$\begin{aligned}(x_r - x_0)^2 + (y_r - y_0)^2 + (z_r - z_0)^2 &= d_{tr}^2 \ \pm C_r \\ (x_{j_1} - x_0)^2 + (y_{j_1} - y_0)^2 + (z_{j_1} - z_0)^2 &= d_{tj_1}^2 \ \pm C_{j_1} \\ (x_{j_2} - x_0)^2 + (y_{j_2} - y_0)^2 + (z_{j_2} - z_0)^2 &= d_{tj_2}^2 \ \pm C_{j_2} \\ (x_{j_3} - x_0)^2 + (y_{j_3} - y_0)^2 + (z_{j_3} - z_0)^2 &= d_{tj_3}^2 \ \pm C_{j_3}\end{aligned} \tag{4}$$

where $|C_i| \leq \theta$.

2) Malicious UAV Detection

There are four possible cases for misbehaving nodes:

1) **Target is malicious.** The target reports a false position.
2) **Jury is malicious.** One or more juries report an incorrect distance about the target to the receiver.
3) **Receiver is malicious.** The receiver can forge/skip all the calculations and report a piece of arbitrary information (e.g., legit, malicious) about the target.
4) **All entities (Target, Jury, Receiver) are malicious.** Multiple nodes lie about positions (e.g., the target and a subset of juries are malicious, and/or the receiver is malicious).

Based on the above scenarios, we introduce the following observation to determine the legitimacy of the target u_t.

Observation 1. *If some jury u_{j_k} claimed an incorrect distance to u_r (i.e., d_{tj_k} is not legitimate) in order to incorrectly show u_t as malicious, then the following expression will hold:*

$$(x_{j_k} - x_t)^2 + (y_{j_k} - y_t)^2 + (z_{j_k} - z_t)^2 - d_{tj_k}^2 = C_k \tag{5}$$

where $|C_k| \geq \theta$.

From the receiver's perspective, it may appear that the target u_t is malicious, while in fact a jury is the malicious entity. This may result in false positives. To mitigate this, GRIFFIN uses a *majority voting* protocol. In particular, GRIFFIN collects the results from all the nodes and makes the final decision. We note that GRIFFIN is designed assuming the worst-case scenario (i.e., all the entities could be malicious). We now present how GRIFFIN employs the majority voting scheme.

C. MAJORITY VOTING & DETECTION RESULT

As mentioned earlier, we cannot confirm a UAV is malicious by observing the results from a single UAV. Likewise, we cannot confirm that a node is legitimate using one calculation. Hence, we propose a majority voting protocol. The voting is conducted by all the UAVs in the swarm. Each UAV will flag other UAVs based on distance calculations (Sec. IV-B1). If all the conditions are satisfied (e.g., Eq. 4), the receiver UAV assigns a non-negative flag (e.g., $+1$). Otherwise, a negative flag (e.g., -1) is assigned. The positive flag indicates that the node is potentially legitimate, whereas the negative flag indicates that it is potentially malicious. Each UAV broadcasts their result (e.g., flag values) within the swarm. Based on the received results, all UAVs assign a Trust Level (TL) to each other UAV. A Trust Level is an integer counter that is used to track how many times a UAV has been marked malicious or legitimate (add $+1$ if legitimate and -1 if malicious). The TL is updated in each round of the algorithm (i.e., on each time slot). When a UAV's Trust Level falls below a predefined negative threshold (e.g., $TL \leq \sigma$, where $\sigma \leq 0$), GRIFFIN finally marks the UAV as malicious and reports this information to the ground station for further post-detection steps.[3] We note that the majority voting used in GRIFFIN differs from classical Byzantine Fault Tolerance (BFT). Classical BFT protocols, such as PBFT, are designed for full consensus and tolerate up to about 33% Byzantine nodes [33]. In contrast, GRIFFIN assumes an honest majority, where malicious votes do not dominate the aggregation. We choose simple majority voting because GRIFFIN only needs a lightweight mechanism to aggregate local detection flags –produced by RSSI-based distance verification and jury-based geometric verification– rather than a full BFT consensus protocol. Therefore, GRIFFIN uses majority voting as a practical and lightweight aggregation step for resource-constrained UAV swarms.

Selecting the Trust-Level Threshold

The threshold σ controls how much negative evidence is needed before a UAV is labeled malicious. Let V^+ be the number of positive votes and V^- be the number of negative votes for a UAV. Its trust level is computed as $TL = V^+ - V^-$. Therefore, the rule $TL \leq \sigma$ means that the UAV is declared malicious only when the negative votes exceed the positive votes by at least $|\sigma|$. Thus, $|\sigma|$ acts as a confirmation margin. A small value such as $|\sigma| = 1$ enables fast detection, but it may react to a single noisy or incorrect vote. A larger value requires repeated negative evidence, which reduces false alarms but may slightly delay detection. In practice, σ is chosen as a small negative value. This allows GRIFFIN to tolerate occasional transient errors while still detecting malicious UAVs quickly. Under the honest-majority assumption, benign UAVs tend to accumulate positive trust, while malicious UAVs accumulate negative trust, so a modest confirmation margin is sufficient to separate them.

V. ALGORITHM AND CORRECTNESS REQUIREMENTS

A. ALGORITHM AND WORKFLOW

Algorithm 1 formally presents the main steps of our distributed misbehavior detection technique. GRIFFIN executes Algorithm 1 periodically at each UAV upon the arrival of each packet.

The first step in our misbehavior detection process is broadcasting the current GPS position (latitude, longitude, altitude) (Line 12). Each UAV then waits for requests to transmit already calculated distances. If a

[3] We refer readers to Sec. VII for possible actions.

Algorithm 1 Misbehavior Detection in GRIFFIN

INPUT
For a packet arriving at time t: *(a)* RSSI measurement, *(b)* GPS coordinates (latitude, longitude, altitude), and *(c)* distances between juries and target
OUTPUT
$TL[u_i]_{\forall u_i \in \mathcal{U}}$: Trust Level of a UAV u_i */* The UAV is tagged malicious if $TL[u_i] < 0$; and marked legitimate otherwise */*
INITIALIZATION
$\mathcal{U} = \{u_1, u_2, ..., u_n\}$
d_{RSSI} distance using RSSI
d_{GPS} distance using GPS
C GPS and RSSI noise error
$\pm\theta$ GPS and RSSI noise acceptable threshold
σ Trust Level threshold for making a decision
BEGIN
Broadcast current position (x, y, z) to all UAVs
Send distances as requested
for each u in $\mathcal{U} - \{u_{self}\}$ **do**
  $RSSI =$ `get_RSSI()`
  $d_{RSSI}(\hat{t}) =$ `RSSI_to_distance()`
  $x_u, y_u, z_u =$ `receive_GPS_position()`
  $d_{GPS}(\hat{t}) =$ `GPS_to_distance()`
  if $|d_{GPS}(\hat{t}) - d_{RSSI}(\hat{t})| > \theta$ **then**
    mark u as malicious ($TL[u] = TL[u] - 1$)
  else
    Randomly select three juries from $\mathcal{U}$: j1, j2, j3
    $x_{j_1}, y_{j_1}, z_{j_1} =$ `receive_GPS_position()`
    $x_{j_2}, y_{j_2}, z_{j_2} =$ `receive_GPS_position()`
    $x_{j_3}, y_{j_3}, z_{j_3} =$ `receive_GPS_position()`
    $d_{uj_1}(\hat{t}) =$ `receive_distance_from_j1_to_u()`
    $d_{uj_2}(\hat{t}) =$ `receive_distance_from_j2_to_u()`
    $d_{uj_3}(\hat{t}) =$ `receive_distance_from_j3_to_u()`
    / Check Eq. 5 using self, juries, and received distances */*
    $(x_r - x_t)^2 + (y_r - y_t)^2 + (z_r - z_t)^2 - d_{tr}^2(\hat{t}) = \pm C_1$
    $(x_{j_1} - x_t)^2 + (y_{j_1} - y_t)^2 + (z_{j_1} - z_t)^2 - d_{tj_1}^2(\hat{t}) = \pm C_2$
    $(x_{j_2} - x_t)^2 + (y_{j_2} - y_t)^2 + (z_{j_2} - z_t)^2 - d_{tj_2}^2(\hat{t}) = \pm C_3$
    $(x_{j_3} - x_t)^2 + (y_{j_3} - y_t)^2 + (z_{j_3} - z_t)^2 - d_{tj_3}^2(\hat{t}) = \pm C_4$
    if $|C_i| \leq \theta$ for all $i \in \{1, 2, 3, 4\}$ **then**
      Mark u as legitimate and update trust score: $TL[u] = TL[u] + 1$
    else
      Mark u as malicious and update trust score: $TL[u] = TL[u] - 1$
    end if
  end if
end for
Broadcast result to the swarm and receive decisions from other UAVs
Perform majority voting
for each u in $\mathcal{U} - \{u_{self}\}$ **do**
  Based on the results, update the Trust Level (TL) for each u:
    • $TL[u] = TL[u] - 1$; if u is malicious (based on voting)
    • $TL[u] = TL[u] + 1$; if u is legitimate (based on voting)
  if $TL[u] < \sigma$ **then**
    Mark u as *malicious* and report to ground station for post detection steps
  end if
end for
END

given UAV is selected as a jury, the receiver (i.e., verifier) node will request the jury to share its distance to the target. We then loop over all the UAVs and *(a)* check the distance calculated using the received GPS position (i.e., using `receive_GPS_position()` and `GPS_to_distance()` functions, respectively), and *(b)* the distance calculated using measured RSSI (i.e., using `get_RSSI()` and `RSSI_to_distance()` functions). We mark the node as malicious if the difference between the two distances is beyond the predefined error threshold (θ). If this is not the case, we continue for the next step. For this phase, the receiver randomly selects three UAVs (except the target) from the swarm as "juries". The receiver stores the 3D GPS positions of the juries (`receive_GPS_position()`) and asks for their distances to the target UAV u (`receive_distance_from_j1_to_u()`, `receive_distance_from_j2_to_u()`, and `receive_distance_from_j3_to_u()`). We then check the 3D sphere constraints in Eq. (4) for the receiver u_r and the three juries u_{j_1}, u_{j_2}, u_{j_3}. If C_1, C_2, C_3, and C_4 are within the defined error range (i.e., $-\theta < C_k < +\theta$), we mark the UAV u as legitimate; otherwise, we mark it as malicious.

B. FORMAL GUARANTEES OF GRIFFIN

Section IV-C presents our majority voting algorithm where we argue that such a voting is required to reach a consensus about the target. We now show that if we have *exactly four 'trusted UAVs'* in the swarm (one receiver and three juries), GRIFFIN obtains similar results without relying on the majority voting process.

Definition 1 (Trusted UAV)**.** *A trusted UAV is a node in the swarm that (a) shares its correct position, (b) claims the correct distance between other UAVs, and (c) faithfully marks other UAVs as either legitimate or malicious.*

Based on our definition of *trusted UAV*, we now present the following theorem:

Theorem 1. *If four trusted UAVs exist in the flock (one receiver and three juries), it is possible to check whether a given target UAV is malicious or not without relying on any majority voting protocol.*

Proof. Let $\mathcal{U} = \{u_r, u_{j1}, u_{j2}, u_{j3}, u_1, u_2, u_3, ..., u_k\}$ define n UAVs present in the swarm ($k < n$), where u_r is the Receiver, and u_{j1}, u_{j2}, u_{j3} are the three juries. Let us assume that these four UAVs are trusted throughout the mission. We denote the set of target UAVs as $u_T = \{u_1, u_2, ..., u_k\}$, where $u_T \subset \mathcal{U}$. Let $[a, b]$ denote the acceptable error interval.

We prove our claim by induction. As we need at least four trusted UAVs (Receiver, Jury1, Jury2, Jury3), our base condition includes these four UAVs, excluding the target UAV. Hence, our base condition contains five UAVs where $k = 1$ ($n = 5$) and $U = \{u_r, u_{j1}, u_{j2}, u_{j3}, u_1\}$.

Using the 3D consistency constraints for u_r, u_{j1}, u_{j2}, u_{j3},

and the target u_1, we obtain:

$$\begin{aligned}(x_r - x_1)^2 + (y_r - y_1)^2 + (z_r - z_1)^2 - d_{tr}^2 &= C_1\\ (x_{j1} - x_1)^2 + (y_{j1} - y_1)^2 + (z_{j1} - z_1)^2 - d_{tj1}^2 &= C_2\\ (x_{j2} - x_1)^2 + (y_{j2} - y_1)^2 + (z_{j2} - z_1)^2 - d_{tj2}^2 &= C_3\\ (x_{j3} - x_1)^2 + (y_{j3} - y_1)^2 + (z_{j3} - z_1)^2 - d_{tj3}^2 &= C_4\end{aligned} \tag{6}$$

If $a < C_1, C_2, C_3, C_4 < b$, then u_1 is legitimate (i.e., u_1 is the common point of the four spheres centered at $u_r, u_{j1}, u_{j2}, u_{j3}$). Otherwise, u_1 is malicious. Let us assume that Eq. (6) is true for n target UAVs, i.e., $u_T = \{u_1, u_2, ..., u_n\}$. We prove that when the condition holds for n target UAVs, it is also correct for the $(n+1)^{\text{th}}$ UAV.

For n targets, we have (for each u_m, $1 \leq m \leq n$):

$$\begin{aligned}(x_r - x_m)^2 + (y_r - y_m)^2 + (z_r - z_m)^2 - d_{tr}^2 &= C_1\\ (x_{j1} - x_m)^2 + (y_{j1} - y_m)^2 + (z_{j1} - z_m)^2 - d_{tj1}^2 &= C_2\\ (x_{j2} - x_m)^2 + (y_{j2} - y_m)^2 + (z_{j2} - z_m)^2 - d_{tj2}^2 &= C_3\\ (x_{j3} - x_m)^2 + (y_{j3} - y_m)^2 + (z_{j3} - z_m)^2 - d_{tj3}^2 &= C_4\end{aligned}$$

For each node we check whether all C_k lie in $[a, b]$ and decide legitimacy accordingly. As *(a)* all the n sets of equations are correct, and *(b)* each set is independent of the others, we can claim that the set of equations for $n+1$ nodes—i.e., the $(n+1)^{\text{th}}$ target UAV—is also correct:

$$\begin{aligned}(x_r - x_{n+1})^2 + (y_r - y_{n+1})^2 + (z_r - z_{n+1})^2 - d_{tr}^2 &= C_1\\ (x_{j1} - x_{n+1})^2 + (y_{j1} - y_{n+1})^2 + (z_{j1} - z_{n+1})^2 - d_{tj1}^2 &= C_2\\ (x_{j2} - x_{n+1})^2 + (y_{j2} - y_{n+1})^2 + (z_{j2} - z_{n+1})^2 - d_{tj2}^2 &= C_3\\ (x_{j3} - x_{n+1})^2 + (y_{j3} - y_{n+1})^2 + (z_{j3} - z_{n+1})^2 - d_{tj3}^2 &= C_4\end{aligned}$$

The above expressions show that if we have *four trusted* UAVs (i.e., Receiver, Jury1, Jury2, and Jury3), we can rely on their calculations and check the positions of all other UAVs without running a majority voting protocol. This concludes the proof. □

VI. EVALUATION

We implement and evaluate GRIFFIN on both *(a)* a realistic UAV simulator (ArduSim [34]) and *(b)* a Raspberry Pi [12] and Navio [13]-based drone platform for runtime overhead.

A. SIMULATION-BASED EVALUATION

1) Simulation Setup

We use ArduSim [11] to evaluate the effectiveness of GRIFFIN. ArduSim is a real-time UAV simulator that instantiates a software-in-the-loop (SITL) environment for each virtual UAV. Each UAV is composed of an agent that controls the vehicle and multiple threads that implement runtime protocols. Inter-UAV communication is based on MAVLink [35]. In our setup, packet exchange is implemented using a sender thread (Talker) and a receiver thread (Listener) at each UAV, and all messages are broadcast to the swarm.

Figure 6 depicts one instance of our simulation setup. We simulate a reconnaissance mission of approximately 8 miles, and evaluate swarm sizes $n \in \{20, 40, 60, 100\}$. The swarm formation is random and UAV speeds are varied in $[10, 40]$ m/s, with altitude in $[10, 30]$ m. The simulation produces 200 time slots, where each time slot corresponds to a packet reception event. The Trust Level counter is reinitialized after each batch, and each batch contains 3 time slots. Since each slot can change the TL by at most ± 1, the TL within one batch is always between -3 and $+3$. Therefore, choosing $\sigma = -3$ is the most conservative option in a batch. A UAV is declared malicious only if it receives negative majority votes in all three slots. This avoids reacting to one noisy vote or a temporary measurement error. Because the counter resets after each batch, the detector does not rely on old evidence for too long and can re-evaluate UAV behavior over time, which is useful against adaptive on–off attackers. We set $\theta = 2$ based on the RSSI noise level and the controlled propagation model used in the simulation. This value provides a balance between rejecting clearly inconsistent position claims and avoiding false alarms caused by measurement noise. A smaller θ makes the detector more sensitive but may increase false positives, whereas a larger θ makes it more permissive and may increase false negatives. From our experiments, we find that the performance of GRIFFIN is independent of the number of UAVs present in the swarm (i.e., 20, 40, 60, and 100 in our setup). Hence, we report the results for a swarm of 40 UAVs.

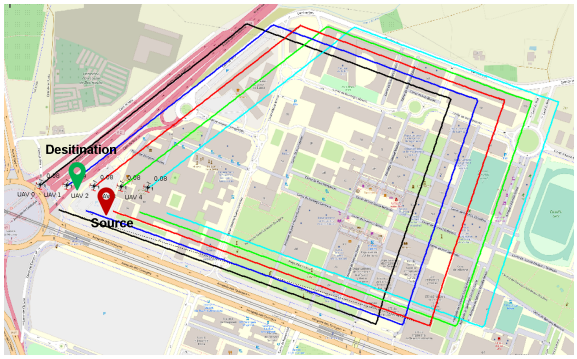


FIGURE 6: Snapshot of an ArduSim reconnaissance mission used in our evaluation.

a: RSSI Modeling in ArduSim

ArduSim does not provide RSSI values directly. Instead, it exposes the ground-truth 3D positions of all UAVs and the MAVLink traffic between them. To obtain RSSI measurements for our framework, we augment ArduSim with an RSSI module that computes a synthetic received power for every packet using the Friis free-space path-loss model at 5 GHz. Concretely, for a transmitter–receiver pair separated by distance d, we model the received power in dBm as: $P_r(dBm) = P_t(dBm) + G_t(dBi) + G_r(dBi) - 20\log_{10}(\frac{4\pi d}{\lambda})$ [26], where P_t is the transmit power, G_t and G_r are the transmit and receive antenna gains, and $\lambda = c/f$ is the carrier wavelength, with $c \approx 3 \times 10^8$ m/s denoting the

speed of light and f the carrier frequency. In our simulations we set $G_t = G_r = 0$ dBi. Table 1 summarizes key simulation parameters.

TABLE 1: Evaluation Setup

Parameter	Value
Number of UAVs, n	20, 40, 60, and 100
Malicious UAVs	$[10\%, 75\%]$ with 5% step
UAV formation	Random
UAV speed	Variable: $[10, 40]$ m/s
UAV altitude	Variable: $[10, 30]$m
RSSI params	$f = 5$ GHz, $P_t = 20$ dBm
RSSI noise std. (σ_{RSSI})	0.5 dB, RSSI floor -100 dBm
GPS/RSSI error threshold (θ)	2
Trust Level threshold (σ)	-3
Number of simulation slots/packets	200

2) Attack Scenarios

We evaluate GRIFFIN under two classes of adversaries: Random attacks and Intelligent attacks. The adversary in Random attacks lies about its position without any prior calculations or considerations. For this type of attack, we argue that only the first phase of GRIFFIN will be enough for detection, therefore, no jury election or jury input is needed. Meanwhile, the adversary in Intelligent attacks will make sure to preserve RSSI consistency with the receiver. However, they introduce geometric inconsistencies as discussed in Sec. IV-B1 to bypass the first phase of GRIFFIN. Thus, we need to use the full pipeline of GRIFFIN for accurate detection. In all cases, malicious UAV identities are selected uniformly at random according to the specified malicious fraction. We define the six attack scenarios as follows. Table 2 summarizes them.

- **Random 1.1 – Malicious receiver falsification:** Malicious receivers intentionally invert the report signs by issuing $+1$ for malicious targets and -1 for benign targets. Benign receivers rely only on RSSI–distance consistency checks, without any jury-based geometric verification.
- **Random 1.2 – No receiver falsification:** All receivers, whether benign or malicious, behave the same way by applying only RSSI consistency checks. In this setting, malicious receivers do not falsify reports and effectively act like benign receivers.
- **Intelligent 2.1 – Malicious receiver falsification with honest juries:** Malicious receivers still falsify their trust updates ($+1$ for malicious targets and -1 for benign targets), while benign receivers execute the full pipeline (RSSI check plus jury-based verification).
- **Intelligent 2.2 – Malicious receiver falsification with lying juries:** Malicious receivers continue to inject inverted reports, and all other receivers run the full RSSI+jury verification algorithm. Malicious juries strategically lie to protect malicious targets and damage benign ones.
- **Intelligent 2.3 – No receiver falsification with lying juries:** All receivers (malicious and benign) run the complete detection algorithm, combining RSSI checks with jury-based verification. Malicious juries lie strategically to shield malicious targets and undermine benign ones.
- **Intelligent 2.4 – No receiver falsification with honest juries:** All receivers execute the full RSSI plus jury verification algorithm honestly. Juries also report honestly in this scenario.

3) Results

Let us define the following four metrics as performance indicators: *(i) True positive (TP):* number of malicious UAVs detected by GRIFFIN; *(ii) True negative (TN):* number of legitimate UAVs marked as legitimate; *(iii) False positive (FP):* number of legitimate UAVs have been marked as malicious; and *(iv) False negative (FN):* number of malicious UAVs marked as legitimate.

Based on the above definitions, we use the following performance metrics to evaluate the detection efficacy of GRIFFIN: *(a)* true positive rate (i.e., detection rate), *(b)* false negative rate, and *(c)* false positive rates. Note that we do not include true negative rates in our evaluation as this is the direct inverse of false positive rates.

a: Detection Rate

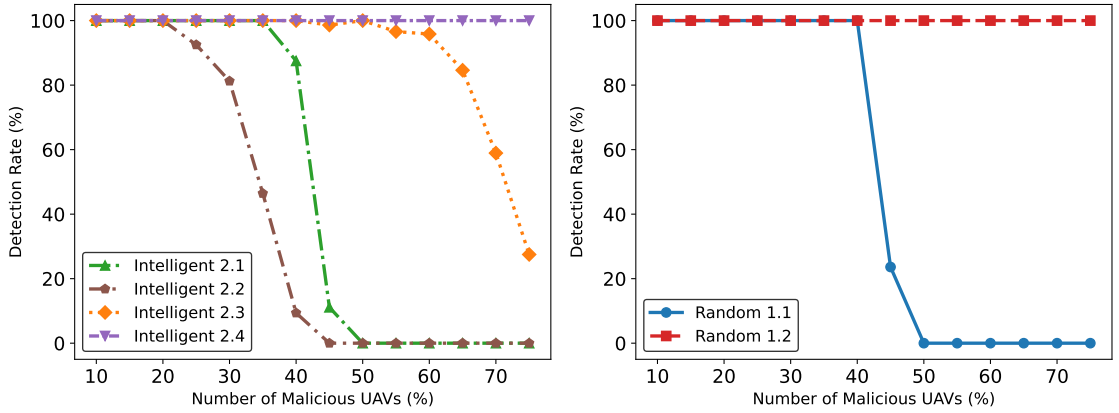
We define the detection rate (i.e., true positive rate) as follows [36]: Detection Rate $(\%) = \frac{TP}{TP+FN}$. We present the detection rate in Fig. 7. The x-axis of Fig. 7 shows the number of malicious UAVs (varied from 10% to 75%), and the y-axis presents the detection rate. In the random attack setting (Fig. 7a), GRIFFIN 's performance is mainly determined by whether receivers falsify their reports: when receivers behave honestly (Scenario 1.2), the RSSI-only consistency check remains highly effective across the swarm, but when malicious receivers actively flip decisions (Scenario 1.1), detection stays strong only while honest receivers still dominate. Detection will collapse sharply once the malicious fraction becomes large enough to bias the aggregated outcome. In the intelligent attack setting (Fig. 7b), adding juries provides stronger verification and can keep detection near-perfect when juries are mostly honest (Scenario 2.4), but performance degrades as jury corruption increases, jury lying typically causes a more gradual decline because it makes verification biased rather than always inverted.

b: False Negative Rate

The false negative rate is the inverse of the detection rate and presented as follows [36]: False Negative Rate $(\%) = \frac{FN}{FN+TP}$. Note that whenever detection remains high, FNR stays near zero, and when detection collapses, FNR rises accordingly. Figure 8 shows the false negative rate obtained from our simulation. In the Random attack setting (Fig. 8a), Scenario 1.2 (the receiver behaves benignly) keeps the false

TABLE 2: Attack scenarios and adversary capabilities.

Adversary class (Target)	Scenario	Receiver is falsifying	Jury is falsifying
Random (Weak)	1.1	✓	N/A
Random (Weak)	1.2	✗	N/A
Intelligent	2.1	✓	✗
Intelligent	2.2	✓	✓
Intelligent	2.3	✗	✓
Intelligent	2.4	✗	✗



(a) (b)

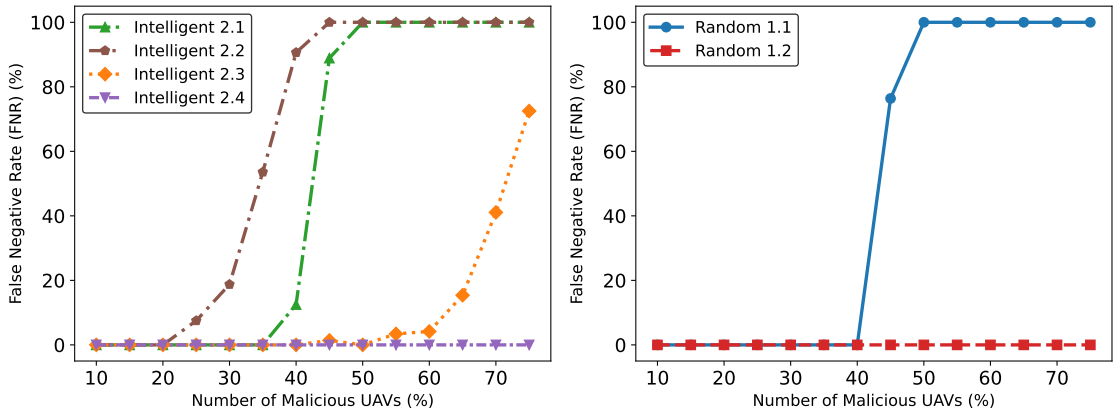
FIGURE 7: Detection Rate (DR) of GRIFFIN under Random vs. intelligent adversaries. *(a)* Intelligent attacks (Scenarios 2.1–2.4), where benign UAVs execute the full GRIFFIN pipeline (RSSI–geometry consistency plus jury verification) and adversaries may additionally influence jury behavior. *(b)* Random attacks (Scenarios 1.1–1.2), where receivers may falsify their reports but benign UAVs perform RSSI-only consistency checks. GRIFFIN sustains near-perfect detection when the decision contributors remain predominantly honest, while detection degrades sharply once adversarial receivers and/or juries become sufficiently prevalent to bias majority decisions.

negative rate low, because the RSSI–distance consistency test repeatedly flags malicious targets in a predictable way. Scenario 1.1 is different: once a malicious receiver starts falsifying its reports, it keeps inverting the decisions protecting the malicious targets and blaming the benign ones, thus, as the malicious fraction increases, FNR rises. In the Intelligent attack setting (Fig. 8b, this effect is even stronger. When receivers falsify (e.g., 2.1/2.2), their false updates make missed detections appear earlier as the number of attackers increases. When receivers do not falsify (e.g., 2.3/2.4), FNR stays lower for longer, because the full GRIFFIN screening keeps producing consistent evidence against malicious targets.

c: False Positive Rate

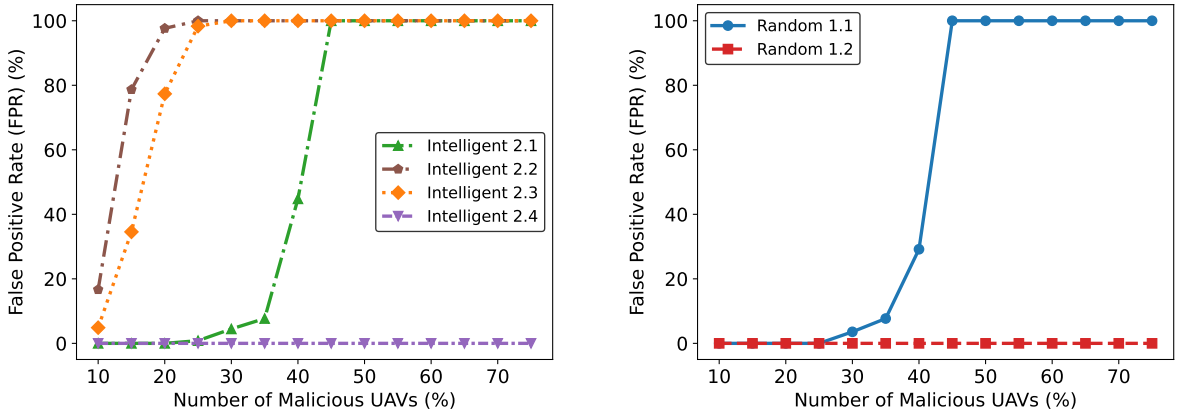
False positive rates tell us how many legitimate UAVs are marked as malicious. We define the the false positive rate as follows [36]: False Positive Rate $(\%) = \frac{FP}{FP+TN}$. In the Random attack class (Fig. 9a), Scenario 1.2 (receiver not falsifying) keeps the FPR essentially at 0% across the entire experiment, which indicates that the RSSI-only consistency test is conservative in the sense that it rarely produces "false alarms" against benign UAVs. In contrast, in Scenario 1.1, FPR stays low at small malicious fractions, then increases sharply (eventually reaching near 100% at higher malicious fractions). This behavior is consistent with receiver falsification act. as the number of compromised UAVs grows, falsified negative updates against benign targets accumulate faster than the swarm can correct them. In the Intelligent attack class (Fig. 9b). Scenarios 2.2 and 2.3 show high false positives even with few attackers, it is because of the lying juries. In Scenario 2.1, this effect is delayed: false positives stay low at first, then rise sharply as the number of malicious UAVs become more influential. Finally, Scenario 2.4 stays close to 0% FPR, reflecting the best-case setting where the detection process remain trustworthy, preventing systematic mislabeling of benign nodes.

Summary of the findings: Overall, GRIFFIN 's behavior is strongly driven by who is allowed to influence the trust update. When receivers behave honestly, the system maintains a high detection rate, which directly translates to low FNR, because the swarm keeps accumulating consistent negative evidence against truly malicious targets. In contrast, whenever receiver falsification is enabled, detection rate drops earlier and more sharply as the malicious fraction increases, and FNR rises correspondingly. On the false



(a) (b)

FIGURE 8: False Negative Rate (FNR) of GRIFFIN under Random (*(b)*) vs. intelligent (*(a)*) adversaries. The figure reports the fraction of malicious UAVs that remain undetected (i.e., labeled benign) as the malicious population increases.



(a) (b)

FIGURE 9: False Positive Rate (FPR) of GRIFFIN under Random (*(b)*) vs. intelligent (*(a)*) adversaries. The plots show how often benign UAVs are incorrectly flagged as malicious as the malicious fraction increases. Scenarios that rely on compromised reports (malicious receivers and/or strategically falsifying juries) exhibit a rapid rise in FPR—often saturating near 100% once adversarial influence becomes dominant—whereas scenarios with honest reporting remain close to 0% across the sweep

positive side. Scenarios where adversaries (receiver and/or juries) can strategically bias the decision tend to increase FPR, because benign UAVs start receiving repeated negative trust updates despite being legitimate. The strongest and most stable operating region is when honest behavior dominates the voting process: you simultaneously see high detection (high TPR), low missed detections (low FNR), and low false alarms (low FPR). As the malicious fraction approaches or exceeds the point where adversarial influence becomes comparable to honest influence, the curves show either the system becomes more conservative (lower FPR but higher FNR) or more aggressive (higher TPR but higher FPR), depending on which attack capabilities are enabled in that scenario.

B. OVERHEAD ANALYSIS ON A REAL UAV

To evaluate the efficacy and overhead of GRIFFIN on a real system, we deploy GRIFFIN on an off-the-shelf drone platform built in our lab. Our UAV prototype (see Fig. 10) is developed using Raspberry Pi (Pi) 4 model B [12] and Navio 2 autopilot hat [13]. The RPi is equipped with a quad core Cortex-A72 (ARM v8) processor and 4GB RAM. We performed a hardware-in-the-loop case study with 100 UAVs where 99 virtual UAVs (simulated) as targets and GRIFFIN runs on one real UAV (i.e., our RPi+Navio-based drone testbed).

We consider the following entities as our performance evaluation metrics: *(a)* execution time overhead, *(b)* memory overhead, and *(c)* CPU utilization. We measure the

FIGURE 10: Raspberry Pi and Navio-based UAV testbed used in our experiments.

execution time using `time()` function in Python [37] . The memory and CPU utilization is measured by using the `memory-profiler` [38] and `psutils` [39] libraries, respectively. From our experiment we find that, on average, GRIFFIN consumes less than 1 MB of memory and less than 1% of CPU usage. Further, the average execution time of GRIFFIN is 2.5 ms. We expect the overheads of GRIFFIN could be acceptable as this extra runtime computations and resource usage (in addition to vanilla operations) result in better security.

VII. DISCUSSION

Our reported results are based on a controlled RSSI model and should be interpreted in light of this assumption. In real UAV-to-UAV communication, RSSI may vary significantly due to multipath propagation, fading, interference, and antenna orientation, which can affect the stability of RSSI-derived distance estimates. As part of our future work, we are extending GRIFFIN to operate under higher-noise propagation settings and evaluating adaptive thresholding mechanisms that remain robust under more realistic RSSI fluctuations. Moreover, the adaptive threshold will take into account other factors such as GPS noise and motion-induced error.

Our experiments show that GRIFFIN works best when the majority of nodes are legitimate. One way to further improve the detection rate is to implement a model that tracks the *history* of each UAV and predicts whether the UAV will be malicious or legitimate in future time slots. The model output can serve as an additional input to our majority voting protocol.

GRIFFIN assumes that all UAVs in the flock must be within communication range. This is a realistic assumption since the nodes need to coordinate and exchange network packets for carrying out the mission objective. However, in many cases (e.g., space exploration or reconnaissance for a large geographic region) the UAVs may not be within the transmission range. In such cases, GRIFFIN can be extended to use "relay UAVs" that can coordinate across swarms. However, ensuring the integrity and trust of relay nodes requires further research.

In the threat model, we assumed that adversaries cannot manipulate the transmit power to change the RSSI values. However, this could be a limitation if a sophisticated attacker can adjust transmit power to produce RSSI values consistent with a fake position. As part of future work, we plan to investigate countermeasures such as monitoring anomalous transmit-power levels and using multiple receivers to detect inconsistent RSSI patterns or suspicious power claims.

In this work, we focus on detecting false position reports. Other parameters include system status, rule specifications, and resource usage that can be fed on top of GRIFFIN to design a generic, multi-objective intrusion detection model. GRIFFIN does not consider the after-effect of detecting an intrusion. Possible actions (after identifying a malicious UAV) include forcing the misbehaving UAVs to return to base or ignoring the communication from (potential) malicious UAVs throughout the mission. However, such schemes require end-to-end coordination and control over UAVs, even after they are compromised or begin to misbehave. Our future work will extend GRIFFIN and design a joint misbehavior detection and response mechanism.

VIII. CONCLUSION

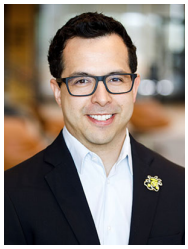
This paper presents a distributed, lightweight misbehavior detection framework for UAV swarms that can detect malicious UAVs, at runtime, without requiring any sophisticated hardware. Our formal guarantees on the number of "trusted" nodes further help the engineers to carefully design their mission objectives. To the best of our knowledge, GRIFFIN is one of the earliest efforts that shows that *it is feasible to detect misbehaving UAVs in a distributed manner, with high efficacy*, using only readily available information such as GPS coordinates and RSSI measurements.

REFERENCES

[1] P. Fahlstrom and T. Gleason, Introduction to UAV systems. John Wiley & Sons, 2012.
[2] A. Saha, A. Kumar, and A. K. Sahu, "Fpv drone with gps used for surveillance in remote areas," in 2017 Third International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), pp. 62–67, IEEE, 2017.
[3] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, "Help from the sky: Leveraging uavs for disaster management," IEEE Pervasive Computing, vol. 16, no. 1, pp. 24–32, 2017.
[4] V. Puri, A. Nayyar, and L. Raja, "Agriculture drones: A modern breakthrough in precision agriculture," Journal of Statistics and Management Systems, vol. 20, no. 4, pp. 507–518, 2017.
[5] J. Schleich, A. Panchapakesan, G. Danoy, and P. Bouvry, "Uav fleet area coverage with network connectivity constraint," in Proceedings of the 11th ACM international symposium on Mobility management and wireless access, pp. 131–138, 2013.
[6] M. R. Brust, G. Danoy, P. Bouvry, D. Gashi, H. Pathak, and M. P. Gonçalves, "Defending against intrusion of malicious uavs with networked uav defense swarms," in 2017 IEEE 42nd Conference on Local Computer Networks Workshops (LCN Workshops), (Singapore), pp. 103–111, 2017.
[7] G. Raja, S. Anbalagan, A. Ganapathisubramaniyan, M. S. Selvakumar, A. K. Bashir, and S. Mumtaz, "Efficient and secured swarm pattern multi-uav communication," IEEE Transactions on Vehicular Technology, vol. 70, pp. 7050–7058, July 2021.
[8] L. M. Da Silva, I. G. Ferrão, C. Dezan, D. Espes, and K. R. L. J. C. Branco, "Anomaly-based intrusion detection system for in-flight and network

security in uav swarm," in 2023 International Conference on Unmanned Aircraft Systems (ICUAS), (Warsaw, Poland), pp. 812–819, 2023.
[9] X. Wang, Z. Zhao, L. Yi, Z. Ning, L. Guo, F. R. Yu, and S. Guo, "A survey on security of uav swarm networks: Attacks and countermeasures," ACM Computing Surveys, vol. 57, no. 3, pp. 1–37, 2024.
[10] D. P. Shepard, J. A. Bhatti, T. E. Humphreys, and A. A. Fansler, "Evaluation of smart grid and civilian uav vulnerability to gps spoofing attacks," in Proceedings of the 25th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2012), pp. 3591–3605, 2012.
[11] F. Fabra, C. T. Calafate, J. C. Cano, and P. Manzoni, "Ardusim: Accurate and real-time multicopter simulation," Simulation Modelling Practice and Theory, vol. 87, pp. 170–190, 2018.
[12] https://www.raspberrypi.com. Accessed: April. 2, 2026.
[13] https://navio2.hipi.io/. Accessed: April. 2, 2026.
[14] X. Wang, Z. Zhao, L. Yi, Z. Ning, L. Guo, F. R. Yu, and S. Guo, "A survey on security of uav swarm networks: Attacks and countermeasures," ACM Computing Surveys, vol. 57, pp. 1–37, Mar. 2025.
[15] T. Abera, R. Bahmani, F. Brasser, A. Ibrahim, A.-R. Sadeghi, and M. Schunter, "Diat: Data integrity attestation for resilient collaboration of autonomous systems.," in NDSS, 2019.
[16] M. A. Lopez, M. Baddeley, W. T. Lunardi, A. Pandey, and J.-P. Giacalone, "Towards secure wireless mesh networks for uav swarm connectivity: Current threats, research, and opportunities," in 2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS), pp. 319–326, IEEE, 2021.
[17] X. Sun, D. W. K. Ng, Z. Ding, Y. Xu, and Z. Zhong, "Physical layer security in uav systems: Challenges and opportunities," IEEE Wireless Communications, vol. 26, no. 5, pp. 40–47, 2019.
[18] A. Y. Javaid, W. Sun, V. K. Devabhaktuni, and M. Alam, "Cyber security threat analysis and modeling of an unmanned aerial vehicle system," in 2012 IEEE Conference on Technologies for Homeland Security (HST), pp. 585–590, IEEE, 2012.
[19] R. Mitchell and R. Chen, "Adaptive intrusion detection of malicious unmanned air vehicles using behavior rule specifications," IEEE transactions on systems, man, and cybernetics: systems, vol. 44, no. 5, pp. 593–604, 2013.
[20] J.-P. Condomines, R. Zhang, and N. Larrieu, "Network intrusion detection system for uav ad-hoc communication: From methodology design to real test validation," Ad Hoc Networks, vol. 90, p. 101759, 2019.
[21] X. Tan, S. Su, Z. Zuo, X. Guo, and X. Sun, "Intrusion detection of uavs based on the deep belief network optimized by pso," Sensors (Basel, Switzerland), vol. 19, 2019.
[22] K. Cengiz, S. Lipsa, R. K. Dash, N. Ivković, and M. Konecki, "A novel intrusion detection system based on artificial neural network and genetic algorithm with a new dimensionality reduction technique for uav communication," IEEE Access, vol. 12, pp. 4925–4937, 2024.
[23] H. Sedjelmaci, S. M. Senouci, and M.-A. Messous, "How to detect cyber-attacks in unmanned aerial vehicles network?," in 2016 IEEE Global Communications Conference (GLOBECOM), pp. 1–6, IEEE, 2016.
[24] S. Bhattacharya and T. Başar, "Game-theoretic analysis of an aerial jamming attack on a uav communication network," in proceedings of the 2010 American control conference, pp. 818–823, IEEE, 2010.
[25] S. Li, F. Shan, J. Liu, M. Coppola, C. De Wagter, and G. C. De Croon, "Onboard ranging-based relative localization and stability for lightweight aerial swarms," IEEE Robotics and Automation Letters, 2025.
[26] J. Xu, W. Liu, F. Lang, Y. Zhang, and C. Wang, "Distance measurement model based on rssi in wsn," Wireless Sensor Network, vol. 2, no. 8, p. 606, 2010.
[27] F. Shang, W. Su, Q. Wang, H. Gao, and Q. Fu, "A location estimation algorithm based on rssi vector similarity degree," International Journal of Distributed Sensor Networks, vol. 10, no. 8, p. 371350, 2014.
[28] R. B. Langley, "The utm grid system," GPS world, vol. 9, no. 2, pp. 46–50, 1998.
[29] D. Cohen, "Precalculus: A problems-oriented approach , cengage learning," tech. rep., ISBN 978-0-534-40212-9, 2004.
[30] O. Ceviz, S. Sen, and P. Sadioglu, "A survey of security in uavs and fanets: Issues, threats, analysis of attacks, and solutions," IEEE Communications Surveys & Tutorials, 2024.
[31] C.-Y. Chen, M. Hasan, and S. Mohan, "Securing real-time internet-of-things," Sensors, vol. 18, no. 12, p. 4356, 2018.
[32] M.-K. Yoon, S. Mohan, J. Choi, M. Christodorescu, and L. Sha, "Learning execution contexts from system call distribution for anomaly detection in smart embedded system," in Proceedings of the Second International Conference on Internet-of-Things Design and Implementation, pp. 191–196, 2017.
[33] M. Castro, B. Liskov, et al., "Practical byzantine fault tolerance," in OsDI, vol. 99, pp. 173–186, 1999.
[34] F. Fabra, C. T. Calafate, J. C. Cano, and P. Manzoni, "Ardusim: Accurate and real-time multicopter simulation," Simulation Modelling Practice and Theory, vol. 87, pp. 170–190, 2018.
[35] https://mavlink.io/. Accessed: April. 1, 2026.
[36] M. Sokolova, N. Japkowicz, and S. Szpakowicz, "Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation," in Australasian joint conference on artificial intelligence, pp. 1015–1021, Springer, 2006.
[37] https://www.python.org/. Accessed: April. 1, 2026.
[38] https://pypi.org/project/memory-profiler/. Accessed: April. 1, 2026.
[39] https://pypi.org/project/psutil/. Accessed: April. 1, 2026.

MOHAMED ANIS AGUIDA Received an MS. degree in computer systems from Ecole nationale Superieur d'Informatique ESI ex-INI, Algiers in 2020, and an MS. degree in computer science from Wichita State University in 2023. Pursuing Ph.D in computer science at Wichita State University since 2021. He has been an Assistant Educator (non-tenure) at Wichita State University since August 2023. His research is about cyber physical system security, UAV swarm security, and trusted distributed systems.



SERGIO A. SALINAS MONROY Dr. Salinas Monroy is an Associate Professor at Wichita State University (WSU). He received the Ph.D. degree in Electrical and Computer Engineering from Mississippi State University in 2015, and the B.S. in Telecommunications Engineering from Jackson State University in 2010. His current research interests include security of cyberphysical systems, including power systems, advanced manufacturing, and unmanned aerial systems.

MONOWAR HASAN Dr. Monowar Hasan is a Computer Science Assistant Professor at Washington State University (WSU). He received his Ph.D. in Computer Science from University of Illinois at Urbana-Champaign (UIUC) in 2020 and an M.S. in Electrical and Computer Engineering in 2015 from University of Manitoba (UM). Dr. Hasan's current research interests include exploring security and resiliency techniques of cyber-physical system domains.