IEEE Open Journal of the Computer Society

Received 11 September 2025; revised 7 November 2025; accepted 7 December 2025. Date of publication 12 December 2025; date of current version 30 December 2025. The review of this article was arranged by Associate Editor Francesco Rundo.

Digital Object Identifier 10.1109/OJCS.2025.3643738

Designing for the Unlikely: Mitigation Against Rare, High-Impact Threats

FRANCESCO SASSI [1], MASSIMO LA MORGIA [1], SUSHIL JAJODIA [2] (Life Fellow, IEEE), LUIGI VINCENZO MANCINI [1], AND ALESSANDRO MEI [1] (Member, IEEE)

[1]Dipartimento di Informatica, Sapienza University of Rome, 00185 Rome, Italy
[2]Center for Secure Information Systems, George Mason University, Fairfax, VA 22030 USA

CORRESPONDING AUTHORS: FRANCESCO SASSI (e-mail: sassi@di.uniroma1.it); MASSIMO LA MORGIA (e-mail: lamorgia@di.uniroma1.it).

"This work was supported in part by the Office of Naval Research under Grant N00014-23-1-2132, in part by the projects: MUR National Recovery and Resilience Plan, SEcurity and RIghts in the CyberSpace SERICS under Grant PE00000014, and in part by through the "Young Researchers 2024-SoE" Program funded by the Italian Ministry of University and Research (MUR)" ST3P under Grant B83C24003210001.
This article has supplementary downloadable material available at https://doi.org/10.1109/OJCS.2025.3643738, provided by the authors.

ABSTRACT Cybersecurity strategies are typically built on risk-based frameworks that prioritize threats according to their perceived likelihood, impact, and the cost of mitigation. Although effective against frequent and high-probability risks, this approach systematically underestimates rare but high-impact events. Such events,though statistically unlikely, can produce catastrophic disruption, as illustrated by the 9/11 attacks or the COVID-19 pandemic. Preventing these events would have required sustained investments that seemed economically or socially unjustifiable until after the disaster struck. This tension between efficiency and resilience exposes a systemic weakness: defenders often prioritize short-term risks over resilience for rare adversarial scenarios. This article argues that it is feasible and necessary to develop novel countermeasures that address rare and high-impact risks without imposing prohibitive costs. Using autonomous drone-based border monitoring as a case study, we examine how attackers can exploit the system by injecting fake events. We propose mitigation strategies that introduce modest inefficiencies, such as random perturbations or suboptimal path choices, which substantially reduce susceptibility to manipulation while adding minimal overhead. We argue for a paradigm shift: from optimizing information systems exclusively for efficiency to designing resilient systems that can withstand rare but high-impact events.

INDEX TERMS Rare events, risk analysis, adversarial attacks, drone-based border monitoring.

I. INTRODUCTION

Cybersecurity systems are often designed under practical constraints, leading to the prioritization of threats based on their perceived likelihood and impact [1], [2]. This risk-based approach typically focuses resources on high-probability threats, while rare events, especially those requiring significant investment to mitigate, are often deprioritized. However, overlooking rare, high-impact events merely because they are statistically unlikely can lead to catastrophic consequences. These events underscore a fundamental tension between risk probability and mitigation cost that is not unique to cybersecurity. Examples of such events include the 9/11 attacks on the Twin Towers and the COVID-19 Pandemic, which are highly challenging and costly to prevent. In particular, the COVID-19 pandemic highlighted the effectiveness of preventive measures, such as the widespread use of FFP2 masks, quarantine enforcement, and global social distancing, in preventing the spread of pandemics. However, adopting such measures in the absence of an active crisis would have incurred steep societal, economic, and psychological costs that most governments were unwilling to bear. The same principle applies in security: defenses against low-probability but high-impact threats are often deprioritized until a catastrophic event forces a reevaluation of the underlying assumptions.

This work argues that developing practical and effective countermeasures against rare but high-impact threats is both feasible and necessary. To illustrate our findings, we present a case study in border monitoring, where a region is patrolled by autonomous drones deployed in a grid. This scenario has gained growing attention, as drones have been proposed for

border surveillance in various contexts, from search and rescue operations at sea [3], to monitoring illegal immigration within the European Border Surveillance System (EUROSUR) [4]. More recently, Europe has announced plans to build a "drone wall" along its eastern border to strengthen protection against potential threats [5]. Each drone monitors a portion of the border, and the drone system as a whole is designed to detect and respond to unauthorized border crossing, even in the presence of cyberattacks that can compromise part of the drone fleet. Almost all drone platforms depend on complex software stacks, including operating systems, communication protocols, and navigation algorithms. Such systems are prone to software vulnerabilities that are routinely discovered even in well-tested code. Therefore, any security solution must assume a basic threat model in which some fraction of drones is affected by a software vulnerability. We assume that these vulnerable drones operate normally until the attacker selectively deactivates them by exploiting their vulnerability. The goal of the attacker is to push the border monitoring system into a state where a *compromised pathway* appears, specifically, a chain of vulnerable drones that, once deactivated, would allow undetected border crossings. Note that in the context of our case study, the existence of a compromised pathway constitutes a rare event, as the probability of this state occurring is intentionally kept very low by the design of the border monitoring system. We perform a risk analysis of the border monitoring system under two scenarios: (i) a static configuration, where drones remain in place, and (ii) a dynamic configuration, where drones move according to a deterministic shortest-path protocol in response to events. The risk analysis indicates that, in both cases, the border monitoring system can be designed to minimize the probability of a compromised pathway occurring. However, in the realistic scenario where an attacker fabricates fake border-crossing events that drones mistakenly perceive as real, we show that only a small number of such events is sufficient to strategically orchestrate a path of vulnerable drones. To implement this strategy, we propose an algorithm that an attacker could employ and assess its effectiveness in three security configurations.

Finally, to mitigate this risk, we propose novel and practical mitigation solutions that significantly hinder the ability of an attacker to form a compromised pathway. We propose a mitigation that is performed at regular intervals and a runtime mitigation that is directly integrated into the drone management protocol. In particular, we devise a runtime path perturbation scheme that significantly increases the effort required by the attacker while preserving performance. In summary, our contributions are as follows:

- *Problem Framing:* We identify a systemic weakness in conventional risk-based cybersecurity frameworks: they undervalue rare but high-impact risks. The article highlights the tension between optimizing for frequent threats and preparing for rare, high-impact events.
- *Conceptual Innovation:* We propose a novel design principle: introducing modest deliberate inefficiencies (e.g., random perturbations) can improve robustness against adversarial manipulation without significantly degrading system performance.
- *Case Study:* We focus on drone-based border monitoring as a concrete case study to exemplify the weaknesses discussed above, illustrate the proposed principles and demonstrate the impact of rare events.
- *Applied Demonstration and Validation:* We apply the above principles to the concrete case study of drone-based border monitoring, illustrating how such inefficiencies substantially hinder attackers' ability to manipulate the system. Experimental results support the claim that these strategies raise the attacker's cost while preserving overall effectiveness. We assess the proposed mitigation strategies, analyzing their effectiveness and operational costs. In particular, the runtime path perturbation significantly improves resilience against rare, high-impact attacks at minimal cost.

Our results show that it is possible to design solutions in which the defender consistently makes a modest investment, such as allocating additional resources or accepting a slight reduction in organizational efficiency, to mitigate extremely rare but high-impact events. For example, if an organization employs a resource optimization scheme, a slightly sub-optimal scheme can be designed which, with only a minor efficiency trade-off, effectively reduces the risk of rare events with substantial destructive potential.

II. DRONE-BASED BORDER MONITORING

To investigate our case study, we adopt the formalization and the protocol proposed by Wolfson et al. [6]. We selected this protocol because it offers an optimal constant-time response $O(1)$ to suspicious events and addresses the potential presence of fake events. According to the formalization by Wolfson et al. , the monitored area is represented as a grid of square cells, consisting of r rows and q columns. Each cell c is monitored by a drone that is responsible for that specific portion of the area. In addition to the $(r \times q)$ drones, there is an extra drone E that can occupy any cell in the grid. This extra drone acts as a reinforcement unit, supporting the management of suspicious events. Suppose that the drone surveilling the cell at position (x, y) of the grid detects a suspicious event in its area. It broadcasts a reinforcement request to all neighboring drones, which is propagated throughout the entire drone system. Consequently, the extra drone receives the reinforcement request, and it travels to the cell where the reinforcement request was initiated. However, considering t as the time needed to traverse a single cell of the grid and d the distance, computed using a shortest path algorithm, between the actual position of E and the cell (x, y), the extra drone E needs $t \times d$ unit of time to reach the cell (x, y) and handle the request. To improve the response time, Wolfson et al. proposed the *Coordinated Path Hop (CPH)* algorithm, which allows serving the reinforcement request in a single time step, thanks to cooperation between the extra drone and the monitoring

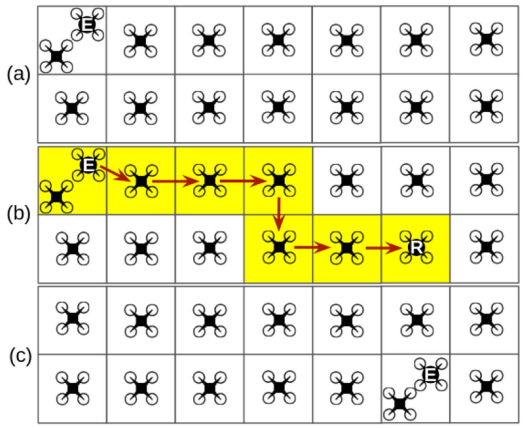

FIGURE 1. (a) Initial grid configuration; (b) request issued with the shortest path highlighted in yellow and drones moving along the path; (c) grid state after serving the request.

drones in handling suspicious events. According to this algorithm, starting from an initial configuration where there is no active request Fig. 1(a), if an event is detected in the cell (x, y) (marked with R in Fig. 1(b), the shortest path (yellow cells) between the extra drone and the request is computed. Then, each drone located along the path between the extra drone and the request location incrementally shifts forward by one cell, following the path. Each red arrow in the Fig. 1(b) represents the movement of an individual drone. The tail of the arrow denotes the current position of the drone, while the head indicates its next position after the movement. The drone that just landed on cell (x, y) becomes the new extra drone and acts as a reinforcement unit, supporting the response to the intrusion event (Fig. 1(c).

III. BORDER MONITORING WITH VULNERABLE DRONES

A. THREAT MODEL AND SECURITY ASSESSMENT SETTINGS

This subsection introduces a threat model for the drone system. In particular, we assume the presence of an organization, referred to as the defender, tasked with monitoring a border for security purposes. To achieve this, the defender deploys a drone-based grid monitoring system. However, a fraction λ of drones are affected by a zero-day vulnerability, and these vulnerable drones are randomly distributed along the border. Although the defender is aware that a fraction λ of drones can be compromised, they do not know which specific drones are affected by a vulnerability. To manage this uncertainty, the defender adopts a risk-based methodology: they estimate the probability that a compromised pathway could form and deploy cost-effective countermeasures until the associated risk is reduced to an acceptable level. From the attacker's perspective, exploiting a drone vulnerability does not give complete control over the drone's movements or functions. Instead, it enables the attacker to disable at least the alarm module, the component responsible for reporting possible unauthorized border crossings. When a drone is disabled, an intruder can pass through the corresponding cell undetected.

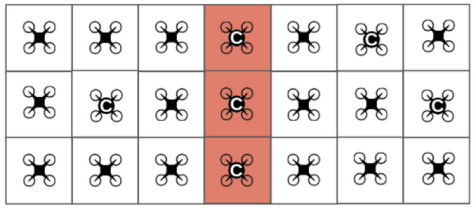

FIGURE 2. An example of *compromised pathway* formed in the border.

The attacker's goal is to enable an unauthorized border crossing while remaining undetected. Thus, his final objective is to compromise a vertical column of vulnerable drones, as indicated in Fig. 2. The figure shows vulnerable drones, marked with a C letter, and an example of a column that is monitored only by vulnerable drones, marked in red. In the following, we refer to this arrangement of drones as *compromised pathway*. If the grid contains a compromised pathway, the attacker can deactivate the alarm module of the drones in the column, enabling an unauthorized crossing. Until this condition is achieved, all drones continue to operate according to the protocol. The threat model is summarized in the following box:

- *Attack vector:* The attacker can disable the alarm module of a fraction λ of drones that have a software vulnerability. If a drone is disabled, an intruder can traverse the cell it monitors without being detected. We assume that the vulnerable drones are randomly distributed within the grid.
- *Win condition:* The goal of the attacker is to create a compromised pathway, *i.e.,* a column of the grid that contains only vulnerable drones.

Sections III-B and III-C illustrate how a defender can evaluate the risk of the monitoring system against the formation of compromised pathways. As described in the threat model, the attacker does not actively interfere with the drone's movements. Consequently, any compromised pathway can only occur spontaneously, with the attacker limited to passive observation to detect if such a pathway has taken place. We perform the risk evaluation in two settings:

- *Static setting (Section III-B):* This setting focuses on estimating the probability that the monitoring system contains a compromised pathway at the time of its deployment. To compute this probability, we assume a static monitoring system in which the drones remain fixed in their initial positions. This setting enables the assessment of the vulnerability of the system without considering the movements of the drones.
- *Dynamic setting (Section III-C):* Then, we examine how many events must occur before a compromised pathway emerges in a dynamic setting. Specifically, the system employs the CPH algorithm described in Subsection II to move the extra drone toward cells where suspicious activity is detected. Here, the attacker exploits the dynamicity of the system by waiting for vulnerable drones to align and form a compromised pathway.

B. RISK ASSESSMENT IN A STATIC SETTING

The first task for a defender is to assess the risk of the system against the defined threat model in a static setting. Assessing the system's risk requires an initial intuitive grasp of the interactions among the various parameters: the number of rows r, the number of columns q, and the fraction of vulnerable drones λ.

- *rows (r):* If the border is monitored by a single line of drones ($r = 1$), a single vulnerable drone is sufficient for an attacker to create a compromised pathway. A countermeasure to mitigate this risk is to increase the number of rows of drones, making r the most critical parameter for the defender. However, increasing r also raises costs, as more drones must be purchased and deployed. It is therefore essential to balance the risk against operational cost.
- *columns (q):* The defender has limited control over the parameter q, which is determined by the length of the border to be monitored and the patrolling capabilities of each drone. Although the defender could intentionally increase the number of columns q, for instance, by underutilizing each drone's monitoring radius, this approach is costly and counterproductive.

A defender can assess the risk of the system by evaluating the probability of compromised pathway formation for given values of r and q, while varying λ. To estimate the probability, the problem can be formulated as follows. Let A be a matrix of size $r \times q$, where r and q denote the number of rows and columns, respectively. Suppose C cells are selected uniformly at random from A, with $C > r$. We are interested in the probability that there exists at least one column in which all r elements have been selected. This probability can be evaluated using the following formula:

$$P\left(\bigcup_{i=1}^{q} A_i\right) = \sum_{k=1}^{\lfloor C/r \rfloor} (-1)^{k+1} \binom{q}{k} \times \frac{\binom{r\times(q-k)}{C-kr}}{\binom{r\times q}{C}}$$

where:

- $\binom{q}{k}$: the number of ways to choose k columns out of q
- $\binom{r\times(q-k)}{C-r}$ is the number of ways to choose the remaining $C - r$ cells from the other $r \times (q - k)$ cells.
- $\lfloor C/r \rfloor$ is the largest integer k such that $kr \leq C$.
- $\binom{r\times q}{C}$ is the number of ways to select C cells out of $r \times q$ cells *i.e.,* all the possible grid configuration when C cells are randomly extracted.

The formula is obtained by first calculating the probability that at least one column is fully compromised and then applying the inclusion-exclusion principle. The complete derivation is provided in the Supplementary Material.

We use this formula to estimate the probability of forming a compromised pathway in the static setting at varying values of r, q, and λ, as shown in Fig 3. The x-axis represents the percentage of compromised drones λ, and the y-axis indicates the probability of forming a compromised pathway. Moreover, each plot shows a different value of q and the dashed lines

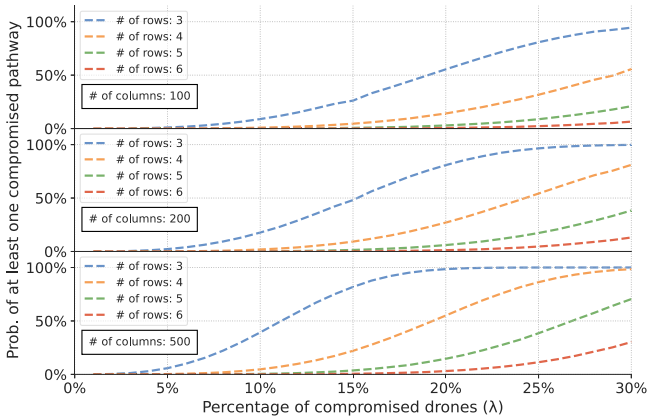

FIGURE 3. Probability of forming a compromised pathway at different values of *r*, *q* and λ in the static setting.

represent the probability of forming a compromised pathway at different numbers of rows r.

As expected, comparing the different lines, we observe that configurations with a higher number of rows (represented in the figures by the red dashed lines) exhibit a significantly lower probability of forming a compromised pathway. Let us consider a configuration with 100 columns and an extreme case where the attacker can compromise 25% of the drones. A matrix with 3 rows (dashed blue line) has an 81% probability of forming a compromised pathway. Increasing the rows to 4 (dashed orange line) reduces the probability to 31% . Finally, with 6 rows, the probability drops further to just 2.2% . An interesting observation emerges when comparing different plots. If we consider the same number of rows and λ as before but increase the number of columns to 200, the probability of forming a compromised pathway increases, reaching 96% for 3 rows, 54% for 4 rows and 4.6% for 6 rows. Thus, increasing the number of columns increases the likelihood of forming a compromised pathway. This is because each column has a certain probability that all its drones are vulnerable. As the number of columns grows, so does the number of chances for such a vulnerable column to appear, even if the individual probability is low. Over many columns, the chance of having at least one fully compromised column increases.

These results highlight that r is the most crucial parameter in determining the system's risk. Since q is largely determined by the border length and the drones' monitoring range, the defender has limited control over it. For the sake of simplicity and comparability, we therefore assume the border can be effectively covered with 200 columns and fix q at this value in the subsequent experiments. We evaluate the risk under three configurations with progressively larger r:

- *Low-security configuration:* $q = 200$, $r = 4$
- *Medium-security configuration:* $q = 200$, $r = 5$
- *High-security configuration:* $q = 200$, $r = 6$

Table 1 shows the results of the formula under the different security configurations (low, medium, and high) for varying values of λ. As expected, higher-security configurations exhibit significantly lower probabilities of compromise across

TABLE 1. Probability of Forming a Compromised Pathway At Varying Values of λ in the Static Setting

λ	Low-security	Medium-security	High-security
1%	0.0004%	$\approx 0\%$	$\approx 0\%$
3%	0.008%	$\approx 0\%$	$\approx 0\%$
5%	0.12%	0.006%	0.0004%
10%	1.9%	0.2%	0.02%
15%	9.2%	1.4%	0.03%

TABLE 2. Average Number of Events for a Compromised Pathway to Form in the Dynamic Setting

λ	Low-security	Medium-security	High-security
1%	862,647	998,160	1,000,000+
3%	39,664	640,618	976,261
5%	5,946	76,761	675,005
10%	313	2,874	25,645
15%	55	389	2,718

all values of λ. While for small fractions of vulnerable drones (e.g., $\lambda = 1\%$ and $\lambda = 3\%$), the probability of forming a compromised pathway is negligible in all security configurations, for higher values of λ, such as 5% and 10% , we observe a clear distinction between the three configurations.

C. RISK ASSESSMENT IN A DYNAMIC SETTING

This section considers a dynamic drone system in which drones respond to real events in accordance with the protocol proposed by Wolfson et al. [6]. As a result, even if the initial deployment of drones on the grid does not form a compromised pathway exploitable by the attacker, such a pathway may emerge over time as drones move to address events within the grid.

As drones move across the grid to respond to events, the attacker continuously monitors whether these responses result in the formation of a compromised pathway. In this case, to assess the risk of the drone system, we built a simulator that completely replicates the protocol and the request handling by the extra drone. The system takes as input the grid dimensions $N = r \times q$ and the fraction of vulnerable drones $0 < \lambda < 1$. It then randomly selects $C = \lfloor r \times q \times \lambda \rfloor$ positions to be occupied by vulnerable drones, while the remaining $N - C$ drones are genuine. Additionally, a random position is assigned to the extra drone, E. The simulation proceeds iteratively until a compromised pathway is formed. In each iteration, a random request position R is generated. The shortest path P from E to R is then computed. If multiple shortest paths exist, one is chosen at random. The drones along P move according to the CPH protocol. If a compromised pathway is formed at any point, the simulation terminates, and the number of events processed is recorded. The pseudo-code for the dynamic simulation is provided in the Supplementary Material.

In this context, we consider the number of events required to form a compromised pathway as a metric to evaluate the system's robustness. Table 2 shows the simulation results in low-security, medium-security, and high-security configurations. The table indicates that the likelihood of a compromised pathway forming is minimal when fewer than 5% of drones are vulnerable. However, this probability increases under low and medium security conditions when 10% to 15% of drones are compromised. Therefore, even though appropriate choices of r can make the occurrence of a compromised pathway exceedingly rare, it cannot be entirely prevented.

IV. MONITORING WITH VULNERABLE DRONES AND FAKE EVENTS

A. THREAT MODEL AND RISK ASSESSMENT SETTINGS

This subsection presents a subtle modification of the previous scenario. The key difference is that the attacker can generate up to $\boldsymbol{K}$ fake events in strategic positions to influence the movement of vulnerable drones within the surveilled area. The threat model is summarized in the following box:

- *Attack Vector:* The attacker can disable the alarm module of a fraction $\boldsymbol{\lambda}$ of drones that have a software vulnerability. Additionally, the attacker can generate $\boldsymbol{K}$ fake events to manipulate the positions of vulnerable drones. We assume that the vulnerable drones are randomly distributed within the grid.
- *Win Condition:* The goal of the attacker is to create a compromised pathway.

In order to assess the risk of the system under this threat model, we devise a strategy that the attacker can use to create a compromised pathway, outlined in Section IV-B). Then, we assess the robustness of the drone system against the proposed attacker strategy in two different settings.

- *Fake events only (Section IV-C):* In this setting, we focus on evaluating the risk of the system under the assumption that the only events it handles are the fake events generated by the attacker. The goal is to assess the average number of fake events required by the attacker to create a compromised pathway.
- *Fake and genuine concurrent events (Section IV-D):* This scenario evaluates the risk of the system considering both fake and genuine events. The attacker uses fake events to strategically manipulate the movement of vulnerable drones, while the system also continues to function with legitimate, naturally occurring events. This mixed setting simulates a more realistic environment where genuine system activity may disrupt the attacker. The assessment focuses on determining how effectively the attacker can still create a compromised pathway despite the presence of genuine events.

B. STRATEGY OF THE ATTACKER

In this subsection, we present a strategy that the attacker can use to create a compromised pathway. First, we describe the procedure for moving the drone one step horizontally to the right. However, the procedures to move the drone

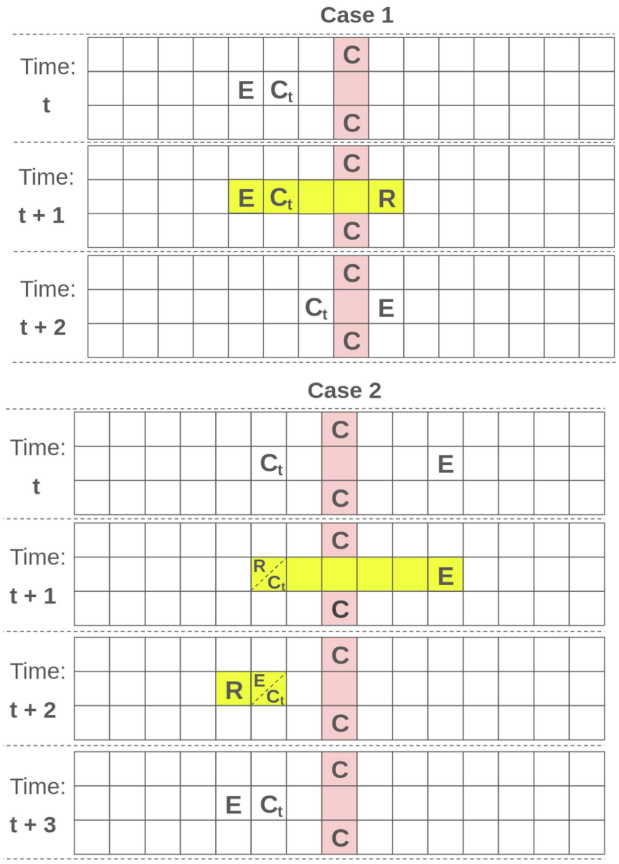

FIGURE 4. An example of application of the MoveRight function, used to move a vulnerable drone to the right.

in the other directions (left, up, or down) follow the same approach, applied symmetrically. Given a targeted vulnerable drone C_t at position (x, y), we provide a function that outputs the positions where the attacker must generate fake events to move C_t to the right $(x, y+1)$. Fig. 4 illustrates two cases where the attacker wants to move the drone toward the target column T (highlighted in red) and the fake events it needs to generate. For a clearer visualization, only relevant drones are shown; however, we assume each cell is monitored by a drone, as depicted in Fig. 1. The figure reports two cases in which the attacker aims to move the vulnerable target drone C_t from its current position (x, y). The two scenarios differ in the position of the extra drone, denoted by E:

- *Case 1:* At time t, the targeted vulnerable drone C_t is located at position (x, y), while the extra drone E is positioned to its left, on the same row. To move C_t toward the target column T, the attacker must generate a fake request R at time $t+1$ in the same row as C_t, but in a column index $y' \geq y+1$ (as shown in the figure). This triggers the extra drone E to move toward R, causing all drones along the computed shortest path (highlighted in yellow) to shift one cell to the right. As a result, at time $t+2$, the vulnerable drone C_t is displaced one step closer to the target column T.
- *Case 2:* In this case, at time t, the attacker cannot immediately move the targeted vulnerable drone C_t to the right using a single fake request, because the extra drone E is positioned to the right of C_t. As a result, any shortest path from E to a fake request placed to the left of C_t would either exclude C_t, leaving it stationary, or, worse, include C_t but shift it leftward—moving it away from the target column rather than toward it.
  To overcome this, the attacker must first reposition E to the left of C_t by generating a fake request in the same cell as C_t, as shown in the figure at time $t+1$. Once E reaches this position, the attacker issues another fake request in a cell to the left of the vulnerable drone—for example, the immediately adjacent left cell, as depicted at time $t+2$.
  By following this approach, at time $t+3$, the extra drone is positioned as in Case 1, allowing the attacker to issue a further fake request that successfully moves C_t one step closer to the target column.

The described strategy is implemented in MoveRight function, detailed in Listing 1. The function returns the position of the fake events that the attacker needs to place to move a vulnerable drone to the right. The procedures to move a drone in the other directions (MoveLeft, MoveUp, and MoveDown) follow the same approach as MoveRight, applied in a symmetric manner. The pseudo-code is provided in the Supplementary Material.

An attacker aiming to create a compromised pathway in column T of the monitored grid can use the following strategy. For each row r in column T:

1) If a vulnerable drone is already present at position (r, T), the attacker skips to the next row.
2) If no vulnerable drone is present in the current row, the attacker computes the Manhattan distance between each vulnerable drone not already in column T and the target position (r, T). The drone minimizing the Manhattan distance is selected to move.
3) If the selected vulnerable drone C_t is not already in row r, the attacker moves it vertically, using MoveDown or MoveUp, to align it with the target row.
4) Once C_t is aligned with row r, the attacker moves it horizontally toward column T using MoveLeft or MoveRight.

Repeating this process for each row allows the attacker to incrementally assemble a full vertical column of vulnerable drones, forming a compromised pathway in column T.

C. EFFICACY OF THE ATTACK IN A SETTING WITH ONLY FAKE EVENTS

In this subsection, we assume that the attacker exploits the vulnerable drones to create a compromised pathway, using the strategy described in the previous subsection. To assess the risk in this setting, we implement the strategy in our simulator and measure the number of fake events the attacker needs to generate in order to create a compromised pathway for different values of λ (1% , 3% , 5% , 10% , and 15%).

Algorithm 1: MOVERIGHT.

Require: E, C
if $E_y < C_y$ **then** ▷ Case 1
  return$\{(C_x, R_y) \mid R_y > C_y\}$
else if $E_y = C_y$ **then** ▷ Case 2: time $t+2$
  $events \leftarrow \{(C_x, R_y) \mid R_y < C_y\}$
  $events \leftarrow events \cup \{(C_x, R_y) \mid R_y > C_y\}$
  return*events*
else if $E_y > C_y$ **then** ▷ Case 2: time t
  $events \leftarrow \{(C_x, C_y)\}$
  $events \leftarrow events \cup \{(C_x, R_y) \mid R_y < C_y\}$
  $events \leftarrow events \cup \{(C_x, R_y) \mid R_y > C_y\}$
  return*events*
end if

TABLE 3. Average Number of Fake Events the Attacker Needs to Create a Compromised Pathway

λ	Low-security	Medium-security	High-security
1%	96.0	126.5	174.8
3%	19.2	31.1	43.2
5%	10.6	16.7	23.3
10%	4.6	7.5	10.7
15%	2.5	4.4	6.5

For each value of λ, the simulator generates 1,000 independent instances of the surveillance grid, randomly placing the fraction λ of vulnerable drones in each instance. In every instance, it computes the number of fake events required to form a compromised pathway in each column, using the strategy described above. It then selects the column that requires the fewest fake events, simulating an attacker who opportunistically chooses the most favorable path. Finally, the simulator reports the average number of fake events over the 1,000 instances for each value of λ.

The results, presented in Table 3, show that with the proposed strategy, the attacker can successfully create a compromised pathway generating a few fake events across all security configurations. Significant differences among the different security levels emerge only for low values of λ (up to 3%). However, with at least 10% of compromised drones, the attacker can create a compromised pathway with fewer than 10 fake events in all security configurations. These findings are particularly impressive compared to Table 2, where the simulator computes the number of events required to form a compromised pathway in a scenario with only genuine events. Indeed, when $\lambda = 1\%$, a compromised pathway may take nearly a million genuine events to form, whereas an attacker using fake events can achieve the same outcome with fewer than 200. This result highlights the limitations of conventional risk-based approaches, which often assume that rare but high-impact events can be considered acceptable risks. Under subtle adversarial manipulation, events considered unlikely become plausible. As a result, defenders cannot rely on perceived rarity as a protective factor, exposing the weakness of risk-based cybersecurity strategies that undervalue rare yet high-impact risks.

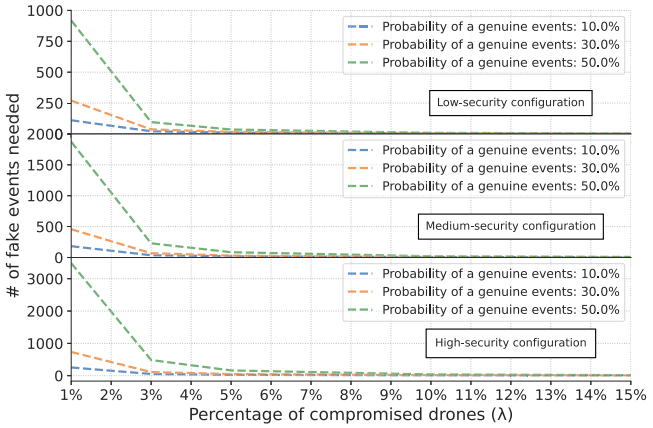

FIGURE 5. Number of fake events needed to form a compromised pathway in the setting where there are both fake and genuine events.

D. EFFICACY OF THE ATTACK WITH FAKE AND GENUINE EVENTS

In this subsection, we evaluate the efficacy of the attack in a scenario where both fake and genuine events can occur. In this setting, the attacker continues to leverage the strategy outlined in Subsection IV-B to manipulate the positions of the vulnerable drones. However, the presence of genuine requests introduce drone movements that disrupt the attacker's strategy: vulnerable drones may be diverted from the target column, and extra drones may end up in unfavorable positions, requiring additional fake events for correction. To simulate this setting we take into account an additional parameter, the frequency of genuine events with respect to the fake ones. For the purpose of the simulation, we assume that, after each fake event, there is a probability $P_{genuine}$ that a genuine event occurs in a a random position of the grid. Like in the previous case, for each security configuration (low, medium, and high), for each value of λ (1% , 3% , 5% , 10% , 15%), and for three values of $P_{genuine}$ (10% , 30% , 50%), the simulator runs 1,000 simulations. As before, we simulate an attacker who opportunistically selects the most favorable column to compromise. For each combination of parameters, the simulator reports the average number of fake events required over the 1,000 instances.

Fig. 5 shows the simulation results, illustrating the number of fake events required for the attacker to create a compromised pathway. Each subplot corresponds to a specific security configuration (low, medium, and high), while the different line colors represent the probability of a genuine event occurring after a fake event: 10% (blue), 30% (orange), and 50% (green). The results reveal notable differences with the setting in which there are only fake events, only for low values of λ, less than 5% .

To illustrate the impact of genuine events on the attacker's strategy, consider a scenario with 10% vulnerable drones and a 30% chance of genuine events. Under these conditions, the number of fake events needed to establish a compromised pathway rises to 6.43, 10.7, and 17.8 for low, medium, and high security configurations, compared to 4.6, 7.5, and 10.7 when genuine events are absent (Table 3). However, if the attacker instead increases the compromised drones to 15% , the required fake events drop to 3.26, 5.93, and 9.73, roughly matching the original targets without genuine events.

Although genuine events may interfere with the attacker's strategy, their impact is negligible, as a compromised pathway can still be created with only slightly more fake events than in the scenario without genuine events.

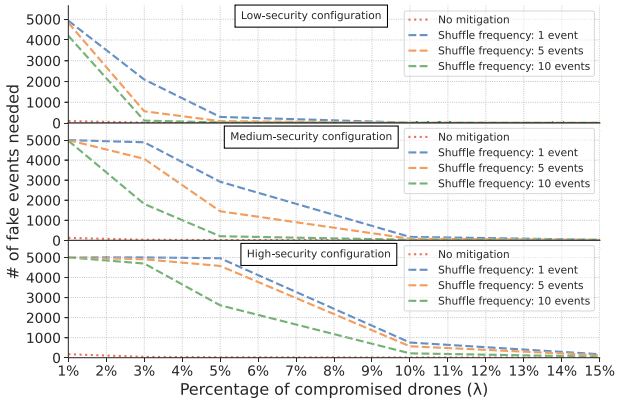

FIGURE 6. Number of fake events needed to form a compromised pathway with Pair Shuffling and without any mitigation strategy.

V. MITIGATION

The previous section shows that a purely risk-based approach, which allows the occurrence of rare events, is inadequate and underscores the need for effective countermeasures. In this section, we provide experimental evidence supporting our conceptual innovation: introducing modest, deliberate inefficiencies to enhance robustness against subtle adversarial manipulation without significantly degrading system performance. In particular, we propose two mitigation strategies that substantially increase the difficulty for an attacker to create a compromised pathway without degrading the system's performance. In the following, we describe the idea behind the two mitigation strategies:

- *Periodic Mitigation:* The idea of shuffling is to periodically relocate all drones within the matrix, in an attempt to disrupt the strategy of the attacker by changing the positions of vulnerable drones. In particular, after every M events, the shuffling redistributes the drones according to a predefined shuffling method.
- *Runtime Mitigation:* This strategy adds randomness to the path between the extra and the request by integrating random perturbations directly into the drone movement protocol.

In the following, we analyze some implementations of these strategies. Then, we perform simulations to evaluate their effectiveness considering the following key metrics:

- *Efficacy:* Measures how many additional fake events an attacker must generate to successfully form a compromised pathway compared to a scenario without mitigation. A highly effective mitigation strategy substantially increases the attacker's required effort, making attacks less practical.
- *Cost:* Represents the operational overhead introduced by a mitigation strategy, defined as the total number of additional steps drones must travel to perform mitigation procedures with respect to the Coordinated Path Hop algorithm described in Sec II. An efficient mitigation strategy keeps this overhead minimal.

By analyzing these factors, we aim to identify the strategy that provides the optimal trade-off between efficacy and cost. To evaluate the mitigation strategies, we conduct simulations that consider only fake events, as the occurrence of genuine events could introduce variability and affect the accuracy of our assessment.

A. PERIODIC MITIGATION: PAIR SHUFFLING

In this strategy, after every M events, drones in the grid are randomly paired, and each drone swaps positions with its assigned partner. This pairing and swapping process continues until all drones have been considered. This random exchange of positions increases unpredictability in drone arrangements, hindering the attacker's ability to create compromised pathways within the grid.

To assess the effectiveness of this strategy, we perform 1,000 simulations for every combination of security configuration, λ (1% , 3% , 5% , 10% , and 15%), and mitigation frequency M (every 2, 5, or 10 events). In this and all subsequent simulations, the process is terminated if a compromised pathway is not reached within 5,000 steps. Fig. 6 shows the efficacy of the Pair Shuffling at varying values of λ. The three plots correspond to the low, medium, and high security configurations, shown from top to bottom. The blue, orange, and green dashed lines indicate the mitigation frequency ($M = 2, 5, 10$), respectively, while the red dotted line shows the number of fake events required to create a compromised pathway without mitigation. The figure shows that frequent mitigation substantially improves efficacy at low values of λ, but its impact diminishes as λ increases.

Table 4 reports the number of fake events required for the attacker to create a compromised pathway under the mitigation strategy (column Pair) for the intermediate case of $M = 5$. While the visualizations above illustrate how the effectiveness of the mitigation strategy varies with different values of M, we report numerical results for the intermediate setting to improve readability and ensure consistency with subsequent experiments. The table shows that, if the percentage of vulnerable drones λ is less than 10% , the number of fake events needed by the attacker increases by at least a factor of 5, demonstrating the effectiveness of the strategy in these

TABLE 4. Average Number of Steps That the Attacker Needs to Create a Compromised Pathway Without Mitigation (—), With Pair, Hamiltonian Shuffling an Path Perturbation in the Different Configuration Settings (Low, Medium, High Security)

	Low Security				Medium Security				High Security			
λ	—	Pair	Hamilt.	Perturb.	—	Pair	Hamilt.	Perturb.	—	Pair	Hamilt.	Perturb.
1%	96	4,825	337	5,000+	126.5	5,000+	740	5,000+	174.8	5000+	2,468	5,000+
3%	19.2	568	58.6	4,776	31.1	4,065	146	5,000+	43.2	4,097	512	5,000+
5%	10.6	91	25.3	4,569	16.7	1,451	78	4,739	23.3	4,576	242	5,000+
10%	4.6	17	11.1	1,231	7.5	84.9	21	3,421	10.7	569	73	4,473
15%	2.5	8.7	7.2	177	4.4	25.5	14	577	6.5	126	39	2,959

TABLE 5. Cost to Apply Mitigation Strategies to Handle 1000 Events

Configuration	Pair	Hamiltonian	Perturbation
Low-security	5.4M	160k	6.3k
Medium-security	6.8M	200k	6.6k
High-security	8.2M	240k	7.1k

scenarios. However, its effectiveness decreases for higher values of λ, particularly in low-security configurations.

Although effective, the Pair Shuffling strategy introduces significant overhead. To evaluate the cost of this mitigation procedure, we simulate a scenario in which the strategy is applied every five events over a sequence of 1,000 events, resulting in 200 mitigation executions. Table 5 reports the total number of steps required in this setting. The results show that the cost of this countermeasure is extremely high, ranging from 5.4 to 8.2 million steps depending on the security configuration. This overhead arises because, during each mitigation phase, all drones are reassigned and may need to travel across distant regions of the surveilled area. One possible way to reduce this overhead is to shuffle only a fraction of the drones. However, this trade-off directly affects the countermeasure's efficacy, as some compromised drones may not be relocated during the mitigation phase.

Additionally, this strategy impacts the response-time optimality guaranteed by the CPH algorithm. Since drones require time to traverse the matrix and complete the shuffle operation, this strategy introduces delays in responding to requests. In the worst-case scenario, if two drones located at opposite ends of the matrix must swap, the cost is $O(r+q)$.

B. PERIODIC MITIGATION: HAMILTONIAN SHUFFLING

The Hamiltonian-based approach is introduced to address the high overhead of the Pair Shuffling strategy. The objective is to design a relocation method in which drones move only one step, consistent with their regular operation during event handling. To this end, we model the monitored border as a grid graph, where each cell is a node connected to its adjacent cells. This graph representation enables shuffling drones by sliding them along Hamiltonian cycles, cyclic paths that visit every node exactly once. In this way, drones relocate in a fully coordinated manner, advancing one step at a time during mitigation and thereby substantially reducing relocation costs.

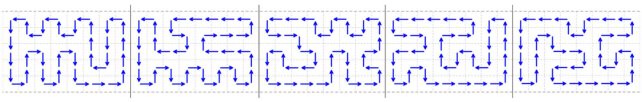

FIGURE 7. This figure illustrates an example of Hamiltonian shuffling applied to a grid of size 5 × 40 with a section of size 5 × 8.

To be effective, this strategy requires varying the Hamiltonian cycle used for shuffling the drone in each mitigation phase. If the same cycle is used repeatedly, it may become predictable to the attacker, who could then adapt their strategy to exploit it. Therefore, an ideal implementation involves computing a different Hamiltonian cycle over the grid graph for each mitigation phase. However, finding a Hamiltonian cycle in a grid graph is NP-hard [7]. Thus, even pre-computing Hamiltonian cycles over a large grid, such as those considered in our study, may be infeasible for the defender.

To address this issue, we propose to divide the grid into smaller sections, for which it is feasible for the defender to precompute the cycle. Then, for each section, all directed Hamiltonian cycles are precomputed. Finally, every M events, a precomputed Hamiltonian cycle is randomly selected for each section, and the drones are shifted along the corresponding path. Fig. 7 shows an example of the application of this strategy on a grid of size 5×40, with chunks of size 5×8. Despite the modest size, the number of such cycles is significant: there are 3,392 different directed Hamiltonian cycles in a 5×8 grid graph.

To evaluate the efficacy of the Hamiltonian Shuffling and allow for a direct comparison with the Pair shuffling method, we conduct simulations using the same experimental settings. Fig. 8 reports the results across varying values of λ, with three plots corresponding to the different security configurations. Dashed lines indicate the mitigation frequency (M), while the red dotted line shows the number of fake events required to create a compromised pathway without any mitigation. The Figure shows that the Hamiltonian-based strategy demonstrates good efficacy, notably increasing the number of fake events required by the attacker, particularly in the Medium and High security configurations. However, its overall effectiveness is lower than that of the Pair shuffling approach (see Table 4).

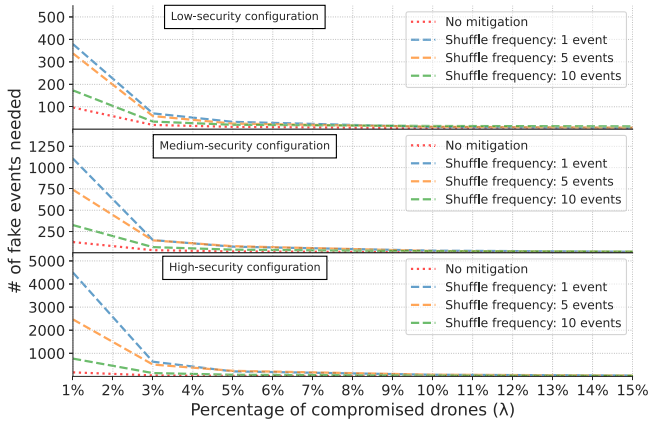

FIGURE 8. Number of fake events needed to form a compromised pathway with Hamiltonian Shuffling and without any mitigation strategy.

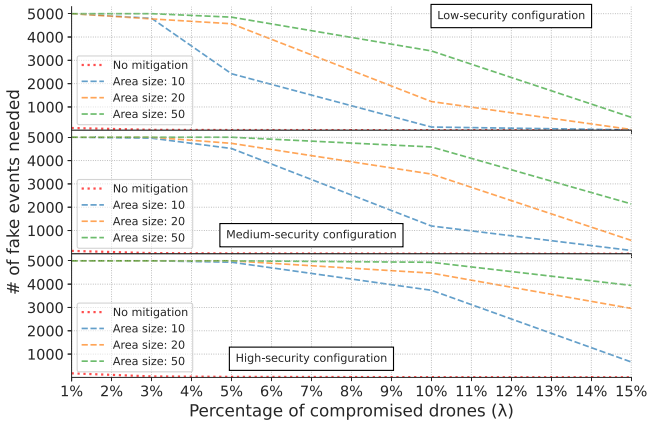

FIGURE 9. Number of fake events needed to form a compromised pathway in absence of any mitigation strategy and with Path Perturbation.

The main limitation of the Hamiltonian approach is that, during each mitigation phase, compromised drones are displaced by at most one cell from their original position. Thus, this strategy is less disruptive than the Pair shuffling strategy, which can reposition compromised drones to distant locations. However, this lower level of disruption offers a substantial cost advantage. As shown in Table 5, the Hamiltonian strategy is far more efficient, requiring over 30 times fewer total steps than the Pair Shuffling technique. Moreover, it causes no degradation in response time, as the entire mitigation completes in a single time step, with all drones moving synchronously by one cell along the predefined Hamiltonian cycle.

C. RUNTIME MITIGATION: PATH PERTURBATION

As the final countermeasure, we introduce the Path Perturbation technique, a runtime mitigation strategy that integrates directly into the drone movement protocol by modifying the Coordinated Path Hop (CPH) algorithm. In CPH, drones respond to an event by moving one step along the shortest path from the position of the extra drone to the event location. Although this approach optimizes for minimal movement cost in responding to events, its deterministic nature makes the resulting drone movements predictable and exploitable by an attacker.

The core idea behind the Path Perturbation technique is to introduce controlled randomness during path computation, slightly altering the shortest path to reduce predictability. In particular, the mitigation introduces a virtual event V, randomly placed within a fixed-size area around the extra drone. Instead of computing the shortest path directly from the extra drone to the request location, the protocol constructs a composite path that first leads to the virtual event V and then continues to the original request. This composite path adds unpredictability to drone movements. Since the attacker cannot anticipate the position of V, they cannot reliably manipulate drone movements toward a compromised pathway, thereby reducing the effectiveness of the attack with only minimal overhead in movement cost.

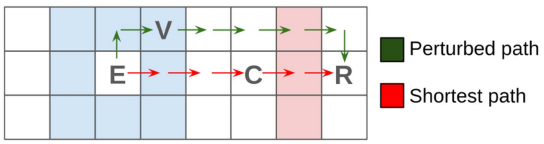

FIGURE 10. Example of the Path perturbation mitigation strategy.

In the following, we propose a procedure for implementing the idea behind the Path Perturbation mitigation.

1) *Define the perturbation area:* Around the position of the extra drone E, define a fixed-size square area of size $l \times l$, referred to as the perturbation area. This region represents the space in which virtual events can be generated to introduce randomness in the path.
2) *Generate a virtual event:* Upon each event request R, uniformly sample a location V within the perturbation area centered on E. This virtual event introduces an intermediate point along the path from E to R.
3) *Construct the perturbed path:* Concatenate the shortest path from E to the virtual event V with the shortest path from V to the original request R, forming a composite path $E \rightarrow V \rightarrow R$.
4) *Move drones along the path:* Apply the same one-cell shift mechanism used in the CPH protocol, moving each drone along the composite path toward the next position, until the request R is fulfilled.

Fig. 10 illustrates an example of the protocol execution. In this scenario, assume the attacker aims to move the vulnerable drone C toward the right in order to form a compromised pathway along the target column (highlighted in red). An extra drone is located at the cell marked with an E. To move C to the right, the attacker places a fake request at position R. Under the standard CPH protocol, the shortest path between the extra drone E and the request location R would be computed (shown with red arrows in the figure). Then, all the drones on the path, including C, slide toward the target column. However, with the path perturbation mitigation in place, a randomized intermediate request V is generated within an area of size

3×3 around the extra drone E, shown in light blue in the figure. Then, the drones follow a perturbed path formed by the concatenation of the shortest path from E to V, and from V to R (shown in green in the figure). This perturbed path prevents the movement of C toward the target column. Since the location of V is very hard to predict for the attacker, he cannot adapt its strategy accordingly.

For the sake of comparison, we evaluate the efficacy of the Path Perturbation under the same settings used for the previous two mitigation strategies. The results of this analysis are shown in Fig. 2. Unlike periodic mitigation strategies, Path Perturbation integrates directly into the drone movement protocol and requires no mitigation frequency M. Instead, it introduces a new parameter: the size of the area $l \times l$ where virtual events may be generated. Accordingly, for each security configuration, we report the number of fake events needed to build a compromised pathway using area sizes of 10×10, 20×20, and 50×50, indicated by blue, orange, and green dashed lines, respectively. In Table 4 we report the results for an area of size 20×20, which empirically provided a good trade-off between the efficacy of the solution and the cost. The table shows that the perturbed path strategy outperforms both the Couple Shuffling and Hamiltonian Shuffling strategies, dramatically increasing the number of fake requests that the attacker needs to perform. Path Perturbation also performs well in the Low Security Scenario, where the Pair and Hamiltonian Shuffling fail to achieve good performances. Indeed, even in a setting with 15% of compromised drones, where the Pair Shuffling and Hamiltonian can increase the number of fake events needed by the attacker from 2.5 to, respectively, 9 and 7 events, the Path Perturbation algorithm achieved excellent performances, bringing the number of events needed to 177. Additionally, as shown in Table 5, the Path perturbation strategy has a much lower cost in terms of steps. In this case, since the mitigation is not a separate dedicated phase, its cost is measured as the extra steps a drone takes when following the perturbed path instead of the shortest path.

VI. GENERAL METHODOLOGY FOR IDENTIFYING AND MITIGATING RARE HIGH-IMPACT CYBER THREATS

This section abstracts the key concepts derived from the drone-based border monitoring case study into a generalizable framework to address rare but high-impact events. The goal is to provide a methodology that can be applied to a wide range of cybersecurity and cyber-physical systems. The framework consists of three core phases that guide the analysis and mitigation process.

Phase 1. System Formalization: The first phase of the framework formalizes the system's operational model by defining its state space, identifying unsafe configurations, and characterizing the transition dynamics that govern its evolution. In particular:

- The **set of all states** $\mathcal{S}$, representing all possible configurations of the system. In the drone monitoring case, $\mathcal{S}$ includes all possible arrangements of drones within the $r \times q$ grid.
- A **subset of unsafe states** $\mathcal{S}_{unsafe} \subseteq \mathcal{S}$, corresponding to configurations that violate safety or security of the system. In our case study, $\mathcal{S}_{unsafe}$ contains all configurations with a *compromised pathway*. This configuration is unsafe, as it allows unauthorized crossing.
- A **set of events** $\mathcal{E}$, representing all events that lead to a change in configuration. In the drone scenario, $\mathcal{E}$ includes all events that cause drones to move on the grid, whether genuine or forged.
- A **transition function** $f : \mathcal{S} \times \mathcal{E} \to \mathcal{S}$, which maps a state and an event to the resulting next state. In our case, f corresponds to the CPH described in Section II, which enables drone movements in response to events.

The first phase establishes the foundation for analyzing the system's behavior. Identifying all possible states, transition functions, and especially unsafe configurations can be challenging and may require a combination of abstract modeling and simulation-based exploration.

Phase 2. Vulnerability Identification: The second phase examines how sequences of adversarial manipulations can exploit the deterministic patterns of the system to trigger catastrophic outcomes. After modeling the system, it is essential to assess whether its transition function reveals deterministic or predictable behaviors that an attacker could exploit. Indeed, even if the probability of reaching an unsafe state is negligible under benign conditions (*i.e.,* it can be considered a rare event), the system's predictability allows an attacker to craft carefully chosen event sequences that gradually steer the system toward $\mathcal{S}_{unsafe}$:

$$S_0 \xrightarrow{E_0} S_1 \xrightarrow{E_1} \dots \xrightarrow{E_{k-1}} S_k,$$

where $S_k \in \mathcal{S}_{unsafe}$. In practical terms, an attacker who can observe or infer the deterministic behavior of f can anticipate the system's reactions and exploit them. Therefore, this phase focuses on identifying predictable transitions, particularly those driven by decision rules that always yield the same optimal response for a given state-event pair.

Phase 3. Mitigation design: The final phase focuses on the introduction of controlled randomness to mitigate vulnerabilities while maintaining acceptable system performance. Randomness can be introduced by periodically altering the state of the system in ways that are unpredictable to the attacker or by modifying the transition function itself to include probabilistic elements. In the drone scenario, the first approach involves periodically shuffling drone positions (periodic mitigation), while the second is achieved through the path perturbation mechanism (runtime perturbation). The simulation results show that the latter preserves near-optimal response times while adding only a minor overhead in the number of drone movements. Thus, a general and practical approach suggested by our case study is to preserve optimal decision-making while introducing slight variations. This strategy ensures that even if attackers understand the system, they cannot reliably predict its evolution, allowing it to continue operating with near-optimal efficiency.

VII. RELATED WORK

Low-probability, high-impact events have been widely studied in economics, where they are commonly referred to as tail risks [8], [9]. They represent statistically rare events in financial markets that often result in disproportionately large losses, such as those experienced during the 2008 financial crisis [10] and COVID-19 [11]. The management of tail risk has been extensively analyzed, with particular attention to its effects on asset prices [12] and strategies for systematic mitigation [13]. However, this kind of events remain largely unexplored in cybersecurity [14], [15], and no prior work addresses drone systems. Our study is the first to analyze this phenomenon and propose countermeasures. We show that deterministic protocols prioritizing optimality, without considering adversarial manipulation, are inherently vulnerable. For instance, Giri et al. [16] enhance Wolfson's grid partitioning algorithm by using non-uniform cells, improves spatial efficiency but remains vulnerable. More generally, our findings apply to any system with optimal path planning [17] in discrete spaces, such as warehouses, smart factories, or autonomous vehicles, where attackers can exploit predictable behavior via fake requests or obstacles, causing unsafe or inefficient configurations.

Black swan events: Black Swan events, popularized by Taleb [18], are highly improbable, extremely consequential, and often rationalized only in retrospect. By definition, they differ from the events we analyze here, which remain are unlikely, but predictable. Aven [19] refines Taleb's view, attributing many such events to the limits of probabilistic reasoning rather than true unpredictability. Lindaas and Pettersen [20] propose "de-blackening" strategies to expose hidden vulnerabilities, while Maslen and Hayes [21] emphasize incident reporting as a way to prevent recurring failures from being misclassified as unforeseeable. Black Swan theory is especially relevant to cybersecurity, where rare attacks can have devastating consequences [22]. Notable examples include the SolarWinds supply chain compromise [23], the Log4J vulnerability [24], and the Colonial Pipeline ransomware attack [25], all of which revealed latent weaknesses with cascading effects across critical infrastructure.

VIII. CONCLUDING REMARKS

This work demonstrates that developing practical mitigation against rare but high-impact threats is both feasible and necessary. Using autonomous drone-based border monitoring as a case study, where events are managed optimally and appear harmless in isolation, attackers can exploit the system's deterministic behavior to trigger deliberately rare but high-impact events. Although the natural formation of a compromised pathway is extremely unlikely, often requiring more than a million random events, our results show that an attacker can achieve the same result with only a few hundred carefully crafted events.

To address these vulnerabilities, we introduced and evaluated two classes of mitigation strategies—periodic and runtime. Our findings show that incorporating small amounts of randomness into the drone coordination protocol substantially strengthens resilience against adversarial manipulation while incurring negligible efficiency costs.

Although our focus was on a specific drone coordination protocol, the underlying lesson is broader: vulnerabilities often arise not from the physical agents themselves, but from deterministic, optimization-driven decision-making in adversarial environments. This insight applies to a wide range of domains, including multi-agent pathfinding, warehouse robotics, and autonomous vehicle routing, where predictability in system behavior can be manipulated to undermine the intended design and safety specification. By embracing modest inefficiencies and randomness, such systems can better withstand rare but high-impact threats without sacrificing operational effectiveness.

REFERENCES

[1] P. Katsumata, J. Hemenway, and W. Gavins, "Cybersecurity risk management," in *Proc. 2010-MILCOM 2010 Mil. Commun. Conf.*, 2010, pp. 890–895.

[2] H. I. Kure, S. Islam, and M. A. Razzaque, "An integrated cyber security risk management approach for a cyber-physical system," *Appl. Sci.*, vol. 8, no. 6, 2018, Art. no. 898.

[3] M. G. Jumbert, "Creating the EU drone: control, sorting, and search and rescue at sea," in *Proc. GoodDrone*. Routledge, 2016, pp. 89–108.

[4] L. Marin, "The humanitarian drone and the borders: Unveiling the rationales underlying the deployment of drones in border surveillance," in *The Future of Drone Use: Opportunities and Threats from Ethical and Legal Perspectives.* The Hague, The Netherlands: T.M.C Asser Press, 2016, pp. 115–132 .

[5] Euronews, "Europe wants to build a drone wall to protect its eastern flank from Russia. is it feasible?," 2025. Accessed: 10 Jul., 2025. Online. [Available]: https://www.euronews.com/next/2025/09/21/europe-wants-to-build-a-drone-wall-to-protect-its-eastern-flank-from-Russia-is-it-feasible

[6] O. Wolfson, P. Giri, S. Jajodia, and G. Trajcevski, "Geographic-region monitoring by drones in adversarial environments," *ACM Trans. Spatial Algorithms Syst.*, vol. 9, no. 3, pp. 1–36, 2023.

[7] A. Itai, C. H. Papadimitriou, and J. L. Szwarcfiter, "Hamilton paths in grid graphs," *SIAM J. Comput.*, vol. 11, no. 4, pp. 676–686, 1982.

[8] V. Bhansali, "Tail risk management," *J. Portfolio Manage.*, vol. 34, no. 4, pp. 68–75, 2008.

[9] G. D. Nicolò and M. Lucchetta, "Forecasting tail risks," *J. Appl. Econometrics*, vol. 32, no. 1, pp. 159–170, 2017.

[10] V. V. Acharya et al. , "Manufacturing tail risk: A perspective on the financial crisis of 2007–2009," *Foundations Trends Finance*, vol. 4, no. 4, pp. 247–325, 2010.

[11] J. Wu, C. Zhang, and Y. Chen, "Analysis of risk correlations among stock markets during the covid-19 pandemic," *Int. Rev. Financial Anal.*, vol. 83, 2022, Art. no. 102220.

[12] B. Kelly and H. Jiang, "Tail risk and asset prices," *Rev. Financial Stud.*, vol. 27, no. 10, pp. 2841–2871, 2014.

[13] M. R. V. Oordt and C. Zhou, "Systematic tail risk," *J. Financial Quantitative Anal.*, vol. 51, no. 2, pp. 685–705, 2016.

[14] T. R. Rakes, J. K. Deane, and L. P. Rees, "It security planning under uncertainty for high-impact events," *Omega*, vol. 40, no. 1, pp. 79–88, 2012.

[15] R. Moreno and G. Strbac, "Integrating high impact low probability events in smart distribution network security standards through cvar optimisation," in *Proc. IET Int. Conf. Resilience Transmiss. Distrib. Netw. (RTDN)*, 2015, pp. 1–6.

[16] P. Giri and G. Trajcevski, "Obstacles aware partitioning for bounding worst case response time of mobile surveillance fleet," in *Proc. IEEE 26th Int. Conf. Mobile Data Manage. (MDM)*, 2025, pp. 1–10.

[17] P. Raja and S. Pugazhenthi, "Optimal path planning of mobile robots: A review," *Int. J. Phys. Sci.*, vol. 7, no. 9, pp. 1314–1320, 2012.

[18] N. Nicholas, "The black swan: The impact of the highly improbable," *J. Manage. Training Institut*, vol. 36, no. 3, pp. 56–57, 2008.

[19] T. Aven, "On the meaning of a black swan in a risk context," *Saf. Sci.*, vol. 57, pp. 44–51, 2013.
[20] O. A. Lindaas and K. A. Pettersen, "Risk analysis and black swans: Two strategies for de-blackening," *J. Risk Res.*, vol. 19, no. 10, pp. 1231–1245, 2016.
[21] S. Maslen and J. Hayes, "Preventing black swans: Incident reporting systems as collective knowledge management," *J. Risk Res.*, vol. 19, no. 10, pp. 1246–1260, 2016.
[22] J. Dykstra and A. Shostack, "Handling pandemic-scale cyber threats: Lessons from COVID-19," in *Proc. New Secur. Paradigms Workshop*, in NSPW '24, New York, NY, USA: Association for Computing Machinery, 2025, pp. 1–10.
[23] R. Alkhadra, J. Abuzaid, M. AlShammari, and N. Mohammad, "Solar winds hack: In-depth analysis and countermeasures," in *Proc. IEEE 12th Int. Conf. Comput. Commun. Netw. Technol. (ICCCNT)*, 2021, pp. 1–7.
[24] D. Everson, L. Cheng, and Z. Zhang, "Log4shell: Redefining the web attack surface," in *Proc. Workshop Meas., Attacks, Defenses Web (MADWeb)*, 2022, pp. 1–8.
[25] J. Beerman, D. Berent, Z. Falter, and S. Bhunia, "A review of colonial pipeline ransomware attack," in *Proc. IEEE/ACM 23rd Int. Symp. Cluster, Cloud Internet Comput. Workshops*, 2023, pp. 8–15.

FRANCESCO SASSI received the M.S. (summa cum laude) degree in computer science and the Ph.D. (summa cum laude) degree in computer science from Sapienza University of Rome, Rome, Italy, in 2020 and 2024, respectively, under the supervision of Prof. Alessandro Mei. He is currently a Postdoctoral Researcher with the Department of Computer Science, Sapienza University of Rome. His research interests include security and privacy, blockchain and cryptocurrencies, and computer systems.

MASSIMO LA MORGIA received the Laurea (summa cum laude) degree in computer science and the Ph.D. degree in computer science from Sapienza University of Rome, Rome, Italy, in 2014 and 2019, respectively, under the supervision of Prof. Alessandro Mei. He is currently an Assistant Professor (RTT) with the Computer Science Department of Sapienza University of Rome. His research interests include computer systems, blockchain, security and privacy, applied machine learning, and computer science and human behavior. He was a Visiting Scholar with the Center for Secure Information Systems of George Mason University, Fairfax, VA, USA, in October and November 2023, hosted by Prof. Sushil Jajodia. In 2023, he was awarded by the European Commission with the Seal of Excellence.

SUSHIL JAJODIA (Life Fellow, IEEE) is currently a University Professor, a BDM International Professor, and the Director of the Center for Secure Information Systems, George Mason University, Fairfax, Virginia. He held permanent positions with NSF, NRL, and the University of Missouri-Columbia, Columbia, MO, USA. He has sustained a highly active research agenda spanning database and cyber security for more than 30 years. He has authored or coauthored seven books, edited 52 books and conference proceedings, and authored or coauthored more than 500 technical articles in refereed journals and conference proceedings. He holds 23 U.S. patents. He was the recipient of the number of prestigious awards in recognition of his research accomplishments. According to Google Scholar, he has more than 56 442 citations, and his H-index is 116.

LUIGI V. MANCINI received the Ph.D. degree in computer science from Newcastle University, Newcastle upon Tyne, U.K., in 1989. He is currently a Full Professor in computer science with the Dipartimento di Informatica, Sapienza University of Rome, Rome, Italy. He has actively participated in program committees for several prominent international conferences in cybersecurity. He recognized the importance of education in cybersecurity and, more than 20 years ago, established a series of master's degree programs in information and network security with the Sapienza University of Rome. He has authored or coauthored more than 140 scientific papers in international conferences and journals.

ALESSANDRO MEI (Member, IEEE) received the Laurea (summa cum laude) degree in computer science from the University of Pisa, Pisa, Italy, in 1994. He is currently a Full Professor with the Department of Computer Science, Sapienza University of Rome, Rome, Italy. He is a Member of the ACM. He was a Marie Curie Fellow from 2010 to 2012. He was a past Associate Editor for IEEE TRANSACTIONS ON COMPUTERS from 2005 to 2009 and the General Chair of IEEE IPDPS 2009, Rome.