



Compartmentalized Resilience Architecture for Mission-Critical Unmanned Aerial Vehicles

Abdel-Rahman Shaout¹, Mohamed Zohdy², & Adnan Shaout³

1. The Electrical and Computer Engineering Department, Oakland University
2. The Electrical and Computer Engineering Department, Oakland University
3. The Electrical and Computer Engineering Department, The University of Michigan - Dearborn

Abstract: Unmanned aerial vehicles (UAVs) increasingly integrate autonomous navigation, wireless communication, artificial intelligence, and modular payloads, increasing the risk that faults or cyberattacks may propagate to flight-critical functions. This paper proposes a compartmentalized resilience architecture for autonomous, network-connected, mission-critical UAVs. It separates flight control, navigation, communication, mission autonomy, payload processing, data management, and safety monitoring into distinct trust domains. A default-deny policy gateway governs inter-compartment communication, while an adaptive trust mechanism restricts or quarantines abnormal components and activates predefined recovery modes. The architecture is assessed through a comparative analytical framework covering representative fault and attack scenarios. The assessment considers containment, flight safety, mission continuity, response time, and anticipated resource overhead. The paper also reviews major UAV applications and their challenges and outlines how compartmentalization can support a modular UAV platform with interchangeable subsystems. The approach aims to limit fault propagation and preserve safe operation while accounting for UAV real-time and resource constraints.

Keywords: UAV security, compartmentalization, cybersecurity, fault containment, resilience, trust management, cyber-physical systems, mission continuity, modular UAV platform

INTRODUCTION

Unmanned aerial vehicles have evolved from remotely controlled aircraft into network-connected cyber-physical systems capable of autonomous navigation, perception, data collection, and mission planning. A UAV, or drone, is commonly defined as an unmanned aircraft or ship guided by remote control or onboard computers [1]. Modern UAVs commonly combine an autopilot, a companion computer, wireless interfaces, positioning and perception sensors, mission-specific payloads, and ground- or cloud-based services. This integration supports infrastructure inspection, logistics, emergency response, defense, precision agriculture, and coordinated multi-UAV operations [38]. Pilotless aircraft were first flown by the United Kingdom in 1917 and by the United States in 1918 [2], [3], and the first modern drone was developed in the United Kingdom in 1935 [4]. Since then, the UAV delivery market has grown rapidly and is projected to continue expanding through 2030 [5]

Greater integration also creates tightly coupled dependencies. A vulnerable payload application may consume resources needed by navigation software. A compromised communication service may submit unauthorized mission commands. Malformed or replayed messages may pass between components, while faulty perception or artificial-intelligence

software may produce unsafe recommendations. In an architecture with broad privileges and communication paths, a local compromise can become a system-level safety failure [39]. Traditional UAV security controls—including encryption, authentication, secure boot, intrusion detection, and redundancy—are necessary but cannot eliminate every compromise. A resilient UAV should therefore assume that some components may fail or become malicious during operation. The consequences depend on whether the architecture can contain the affected component and preserve essential flight functions.

Compartmentalization divides a system into isolated domains with narrowly defined privileges and communication paths. Although established in operating systems and high-assurance computing, applying it to UAVs presents distinctive challenges. UAVs are constrained by processing capacity, memory, weight, energy, bandwidth, and real-time requirements. Isolation that introduces excessive latency or resource consumption could itself undermine safety [40][41].

Full compartmentalization is also inappropriate for every UAV. Toy drones, disposable aircraft, and micro- or nano-UAVs may lack the resources and complexity needed to justify strong isolation. This paper focuses on software-rich, autonomous, network-connected, mission-critical UAVs with sufficient onboard computing resources. Relevant applications include infrastructure inspection, logistics, defense, emergency response, and coordinated multi-UAV operations. The paper makes six contributions:

1. It identifies the UAV classes and applications for which compartmentalization is appropriate.
2. It develops a threat and failure model covering cyberattacks, component faults, and resource interference.
3. It proposes a conceptual compartmentalized architecture for mission-critical UAVs.
4. It defines controlled communication, trust management, quarantine, and degraded-operation mechanisms.
5. It presents an analytical methodology for comparing the proposal with a conventional integrated architecture.
6. It outlines how compartmentalization can support a modular UAV platform with interchangeable subsystems.

The remainder of this paper is organized as follows. Section 2 provides the background, scope, motivation and related work. Section 3 defines the system, threat, and failure models. Section 4 presents the architecture. Section 5 explains the research methodology. Section 6 provides the analytical evaluation and discussion. Section 7 identifies limitations and future work, and Section 8 concludes the paper.

UAV BACKGROUND, APPLICATION SCOPE, AND MOTIVATION

UAV System Architecture

A representative autonomous UAV contains two principal computing layers. The flight-control unit performs stabilization, state estimation, actuator control, and basic fail-safe functions. A companion computer performs computationally intensive tasks such as perception, path planning, communication, and payload processing. The UAV may also

communicate with a ground station, fleet-management platform, cloud service, or other UAVs. This division provides some physical separation but does not necessarily establish complete compartmentalization. Several companion-computer applications may share an operating-system kernel, privileges, and resources. The flight controller may accept high-level commands without adequate validation of their origin, freshness, or safety. Shared processors, memory, and communication channels may also allow one component to interfere indirectly with another.

UAV Applications and Compartmentalization Needs

UAVs have become an established tool across a wide range of civil and military fields [6]. Compartmentalization is most applicable when a UAV has several of the following characteristics:

- Autonomous or semi-autonomous operation;
- Multiple software components and trust levels;
- Modular or third-party payloads;
- External wireless connectivity;
- Sensitive mission data;
- High safety or mission criticality;
- Regular software updates;
- Operation in hostile or uncertain environments; and
- Sufficient processing, memory, and energy capacity.

The applications below differ in how many of these characteristics they exhibit, and therefore in the degree of separation they require.

Recreation and Hobby Use

The FAA defines recreational UAV operation as flying for enjoyment rather than for work [7]. According to FAA data accessed in October 2026, 873,450 drones were registered in the United States, of which 502,105 (approximately 57%) were registered for recreational use [8]. Recreational uses include racing, photography, aerial sightseeing, and entertainment shows. Recreational UAVs are typically low-cost and resource-constrained. They may therefore support only lightweight separation, such as isolating communication and payload functions from flight control.

Emergency Response

UAVs have become an important component of search-and-rescue, firefighting, medical support, and law-enforcement operations. They shorten response times and reach areas that are difficult or dangerous for personnel [6], [9]. These missions depend on live video, communication with responders, and operation in uncertain environments. Emergency-

response UAVs therefore benefit from isolating video analytics and external communication from navigation and safety services.

Delivery

UAV delivery has expanded beyond local and online retail. The concept of a delivered good now includes data as well as physical items, such as traffic information, network coverage, and pandemic tracking [10]. Commercial pressure toward speed and low cost has shaped UAV design, in some cases limiting capabilities or delaying advancement [11]. These limitations reduce confidence in UAVs for some emergency applications.

Military programs have advanced UAV operation in otherwise unreachable areas, but much of that technology has not transferred to civilian use because of restrictions between the two domains. Military research has nonetheless informed civil delivery services. Table 1 summarizes early pioneering programs in UAV delivery, along with their drone models and specifications; several of these programs have since been redesigned or discontinued [36].

Delivery UAVs handle customer and package data alongside flight-critical functions. This makes them natural candidates for separating customer- and package-management functions from flight control.

Table 1: Early pioneers in UAV delivery

Organization	Model name	Supplier/Designer	Specs	Image
Amazon [29] [30]	Prime Air	Amazon's Prime Air	Carry up to 5 lbs 30 minute flight time or less	 [29]
FedEx/Alphabet [31]	Wing A1400	FedEx Institute of Technology/Wing	Carries up to 2.6 lbs Flight range 12 miles [32]	 [33]
UPS [34]	Matternet M2 quadcopter	Flight Forward/Matternet	Carries up to 4.4 lbs Flight range 12 miles	 [34]
DHL [35]	MD4-3000	Microdrones	Carries up to 7.5 lbs Flight range about 8 miles [36]	 [35]

Agriculture

Agricultural UAVs use sensors and cameras to collect the information farmers need for crop-management decisions. Figure 1 summarizes their main uses:

- a. **Harvest monitoring:** Monitoring crops in person is costly and time-consuming. UAV-mounted sensors allow growers to determine harvest timing remotely, reducing labor and cost [13].
- b. **Mapping:** Before planting, farmers must assess soil condition, water absorption, fertilizer levels, and other factors, and this testing is expensive. A mapping UAV can survey the field and provide much of this information, helping determine what and when to plant at lower cost [13], [14].
- c. **Spraying:** UAVs can also be used to apply treatments to crops [13], [14].

- d. Sensing: Low-altitude sensing allows an entire field to be monitored without an expensive fixed monitoring system. Even coverage across the field makes this approach more accurate [15].

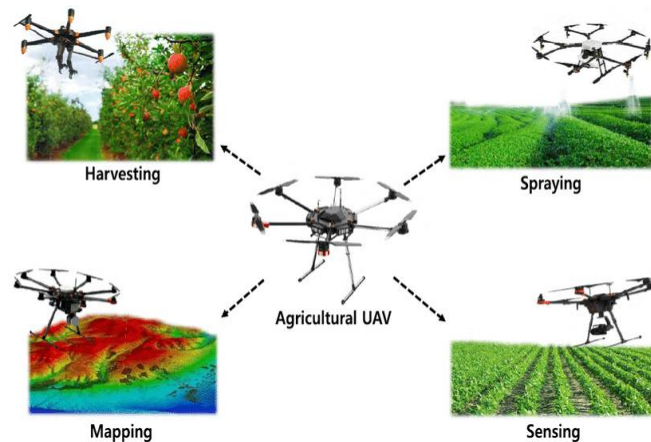


Figure 1: Agricultural uses of UAVs [15]

Adoption nonetheless remains limited. Short endurance, small coverage area, low payload capacity, and weather sensitivity make it difficult for agricultural UAVs to complete tasks efficiently. Despite their low cost, they are often less practical than manned aircraft or satellite systems [9], [12], [13]. Agricultural UAVs typically carry modular sensor payloads but have limited onboard resources, so lightweight payload isolation is the most realistic level of separation for them.

Wireless Network Coverage

UAVs can be costly to develop, build, and maintain. Even so, using them to provide or extend wireless coverage can be more cost-effective, and in some cases more reliable, than cell towers or satellites [17].

Other aerial platforms can serve the same role. High-Altitude Platforms (HAPs), such as balloons, can cover areas spanning thousands of kilometers. They can serve as long-term or even permanent solutions, for example by maintaining a secure network during infrastructure repairs. Low-Altitude Platforms (LAPs) cover tens of kilometers for short periods and were originally intended as short-term solutions, typically for military use [9], [17].

Figure 2 shows three major roles for UAV-aided wireless networks:

- Ubiquitous coverage, in which UAVs extend an existing network during outages or congestion;
- Relaying, in which UAVs provide a secure channel between widely separated users [17]; and
- Data collection, in which UAVs gather data from multiple sources in a single pass. This avoids the delays and corruption that can occur during network outages or heavy traffic [9].

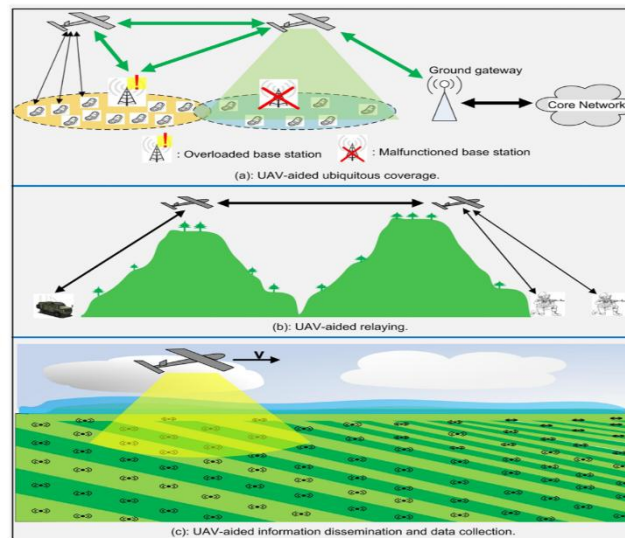


Figure 2: Cases of UAV aided wireless coverages [9] [17]

Because these UAVs act as network infrastructure, they carry third-party traffic and maintain persistent external links. A compromise of the communication stack must not be able to propagate to flight control. Strong separation between networking and flight functions is therefore essential.

Military, Infrastructure, Multi-UAV, and Advanced Air-Mobility Systems

Military UAVs are strong candidates for compartmentalization because they operate in adversarial environments and require separation among communication, navigation, sensing, and mission functions. Critical-infrastructure inspection UAVs combine autonomy, sensitive facility data, specialized sensors, and remote connectivity. Multi-UAV systems benefit from both internal isolation and network-level containment of a compromised member. Advanced air-mobility platforms may require the strongest separation because of their safety and certification requirements [37].

Security and Resilience Motivation

Preventive security attempts to stop unauthorized access or modification. Resilience assumes that prevention may fail and seeks to preserve essential operation. Compartmentalization primarily improves resilience by limiting the authority and propagation potential of a compromised component. The proposal follows four principles:

1. Least privilege: each component receives only the permissions required for its function.
2. Default-deny: communication is blocked unless explicitly permitted.
3. Mediated interaction: cross-compartment messages undergo validation and policy enforcement.
4. Controlled degradation: failure of a nonessential compartment activates a predefined safe response rather than causing uncontrolled system failure.

Compartmentalization does not replace secure communication, software assurance, redundancy, or sensor validation. It provides a structural layer that constrains the consequences of failures in those mechanisms.

Related Work

UAV security. Surveys of UAV security catalog threats across communication links, sensors, onboard software, and ground infrastructure, along with corresponding countermeasures [24], [47]. GNSS spoofing has been demonstrated against operational UAVs [45], and detection approaches rely on consistency checks against independent references [46]. Broader reviews of UAV challenges and avionics architectures likewise identify cybersecurity as a central concern as UAVs become more autonomous and connected [23], [38]. Most of this work focuses on preventing or detecting attacks at the UAV's external interfaces, through secure communication, authentication, and intrusion or spoofing detection. It gives less attention to limiting what a compromised internal component can do.

Partitioning and separation kernels. The principle of dividing a system into isolated partitions that interact only through controlled channels originates with the separation kernel [48]. Crewed aviation applies this principle through integrated modular avionics and the ARINC 653 standard, which provides time and space partitioning so that applications of different criticality can share hardware. Jo et al. implemented ARINC 653 temporal partitioning for small civilian UAVs [50], and Farrukh and West revisited integrated modular avionics for autonomous multicopters [51]. The DARPA HACMS program demonstrated that the formally verified seL4 microkernel could isolate software components on autonomous air vehicles. In red-team testing, attackers given access to a mission-software partition were unable to disrupt flight-critical functions [49].

Compartmentalization of embedded and flight software. Several techniques compartmentalize microcontroller firmware, including automatic compartment generation [52], customized memory views for real-time systems [40], and low-cost compile-time privilege separation [41]. An architectural analysis of NASA's Core Flight System found that broadly shared authority allows a single compromised component to act in ways that are difficult to distinguish from legitimate behavior, which motivates stronger internal trust boundaries in flight software [39].

Runtime safety assurance. The Simplex architecture pairs a complex, high-performance controller with a simple, verified safety controller that takes over when the system approaches an unsafe state [53]. The same idea underlies the independent safety validation in Section 4.4, where mission and artificial-intelligence outputs are treated as recommendations, not commands.

Positioning of this work. This paper builds on these lines of research but differs in emphasis. Instead of proposing a new isolation mechanism, it defines a technology-neutral architecture for mission-critical UAVs. The architecture combines functional compartmentalization, policy-mediated communication, adaptive trust with graduated containment, independent safety validation, and predefined degraded operation, and the paper evaluates it analytically across representative fault and attack scenarios. The architecture can be realized with any of the isolation mechanisms above, from separation kernels and ARINC 653 partitioning to embedded compartmentalization techniques.

SYSTEM, THREAT, AND FAILURE MODELS

System Model

The UAV is modeled as

$$U = \{F, N, C, M, P, D, S\}, \quad (1)$$

where F denotes flight control and stabilization, N navigation and state estimation, C communication, M mission planning and autonomy, P perception and payload processing, D data management, and S safety monitoring and recovery.

Each function has distinct timing, privilege, trust, exposure, and data requirements. Flight control requires predictable execution and direct actuator access. Communication and payload functions are more exposed because they process external or third-party information. Mission and perception components may be complex and difficult to verify but can influence vehicle behavior. The safety function supervises the system and initiates degraded modes.

Security and Safety Objectives

The proposed architecture seeks to:

- Prevent unauthorized access to actuators and flight-critical interfaces;
- Limit fault and attack propagation to the originating compartment;
- Authenticate and validate inter-compartment communication;
- Preserve flight-control timing during abnormal resource consumption;
- Detect malformed, stale, excessive, or unauthorized messages;
- Restrict or quarantine abnormal components;
- Maintain safe degraded operation when mission functions fail; and
- Preserve diagnostic information without disrupting real-time operation.

Threat Model

The study considers an adversary who compromises an externally exposed or lower-trust component, such as a communication service, mission application, payload process, or perception model. The adversary may attempt to send unauthorized commands, modify or fabricate messages, replay stale messages, flood communication channels, exhaust resources, access sensitive data, influence navigation decisions, or move laterally toward flight-critical functions.

The model also includes non-malicious failures such as process crashes, software defects, delayed data, sensor errors, and communication loss. Because these failures can have consequences similar to deliberate attacks, they use the same containment and recovery framework.

The minimal flight controller, isolation mechanism, policy core, and safety monitor are initially assumed to be trusted. Physical destruction, hardware implants, total

compromise of the trusted computing base, and simultaneous compromise of all redundant safety functions are outside the scope.

Failure-Propagation Model

The UAV is represented as a directed dependency graph

$$G = (V, E), \quad (2)$$

where each vertex $v_i \in V$ represents a component and each edge $e_{ij} \in E$ represents a communication, control, or resource dependency. A conventional architecture may contain broad or undocumented edges. The proposed architecture removes unnecessary edges and places a policy decision point on each retained cross-compartment path.

A failure originating in v_i is considered contained when it produces neither unauthorized state changes in other compartments nor violations of safety constraints. The mission may still be interrupted; fault containment, safe recovery, and mission completion are separate outcomes.

ARCHITECTURAL DESIGN AND OPERATIONAL MODEL

This section presents a technology-neutral compartmentalization architecture that isolates exposed and lower-assurance UAV software from flight-critical functions, mediates all cross-compartment interaction, and defines how the system contains and responds to compromised components.

Design Principles

The architecture is based on five design principles that refine those of Section 2.3. Mediated interaction is made concrete through default-deny communication, and two principles are added: criticality separation and independent safety validation. The five principles are:

- **Least privilege:** compartments receive only necessary permissions.
- **Default-deny communication:** interactions require explicit authorization.
- **Criticality separation:** exposed or complex software is isolated from flight-critical functions.
- **Independent safety validation:** mission and artificial-intelligence outputs are checked before influencing flight.
- **Controlled degradation:** a failure triggers an application-specific safe mode.

These principles describe the required behavior without prescribing a specific processor, operating system, or isolation technology.

Functional Decomposition

The UAV's functions form seven conceptual compartments:

1. **Flight control:** stabilization, actuator control, and essential flight modes.

2. **Navigation:** positioning, state estimation, and sensor fusion.
3. **Communication:** ground-control, radio, cellular, satellite, or inter-UAV links.
4. **Mission autonomy:** route planning, waypoint generation, and autonomous decisions.
5. **Payload and perception:** cameras, inspection sensors, object detection, and mission equipment.
6. **Data management:** telemetry, mission data, logs, and sensitive records.
7. **Safety monitoring:** health assessment, policy supervision, and recovery selection.

Assignment is based on safety criticality, exposure to untrusted input, privileges, data sensitivity, and failure-propagation consequences. Flight control receives the highest criticality classification. Communication and payload functions are lower-trust domains because they process external or third-party inputs.

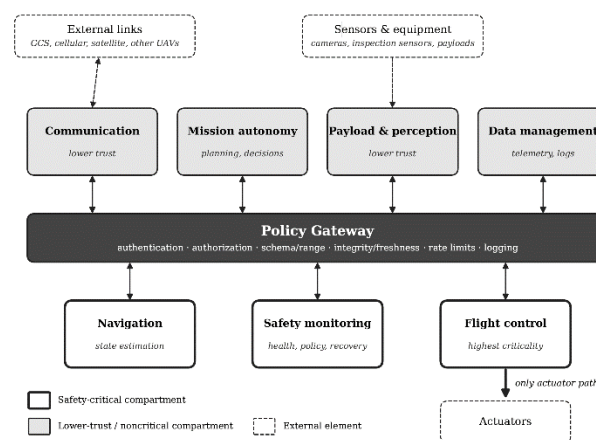


Figure 3: Conceptual compartmentalized UAV architecture.

Figure 3 shows the seven compartments, with all cross-compartment flows passing through the policy gateway and only the flight-control compartment connected to actuators.

Inter-Compartment Communication

Each authorized interaction is represented as

$$R_{ij} = (K_i, K_j, m, r, f, \Delta t, a), \quad (3)$$

where K_i and K_j are the source and destination compartments, m is the permitted message type, r is the valid range, f is the maximum frequency, Δt is the freshness requirement, and a is the violation response.

The policy gateway applies the following:

- Source and destination authentication;
- Message-type authorization;
- Schema, size, and range validation;
- Integrity and freshness checking;

- Rate and resource limits; and
- Security-event recording.

Mission and perception compartments cannot issue actuator commands directly. They submit bounded, high-level recommendations that the flight-control and safety functions validate. This reduces the ability of complex or lower-assurance software to control safety-critical behavior.

Safety Constraints

A flight-related request is accepted only when

$$V(m) = V_{id} \wedge V_{schema} \wedge V_{fresh} \wedge V_{range} \wedge V_{safety}. \quad (4)$$

The safety predicate can include geofence boundaries, altitude and speed limits, obstacle clearance, battery reserves, trajectory constraints, and valid flight-mode transitions.

Authentication alone cannot establish operational safety. It identifies the source but does not prove that the requested action is appropriate. Independent safety validation is therefore required even for authenticated commands.

Trust and Containment Model

Each noncritical compartment receives a trust score:

$$T_i(t) = w_A A_i(t) + w_I I_i(t) + w_B B_i(t) + w_\Phi \Phi_i(t), \quad (5)$$

where A_i represents authentication status, I_i integrity, B_i behavioral conformity, and Φ_i freshness and reliability. The weights satisfy

$$w_A + w_I + w_B + w_\Phi = 1. \quad (6)$$

The compartment state is

$$State_i = \begin{cases} Normal, & T_i \geq \tau_H, \\ Restricted, & \tau_L \leq T_i < \tau_H, \\ Quarantined, & T_i < \tau_L. \end{cases} \quad (7)$$

A restricted compartment may operate with lower message rates or narrower privileges. A quarantined compartment cannot influence safety-critical functions. Severe deterministic violations, such as attempted direct actuator access, trigger immediate quarantine rather than waiting for the trust score to decline.

The trust model provides a conceptual basis for graduated response. Selection of weights and thresholds is application-dependent and would require calibration and validation before operational use.

Degraded Operation

Each compartment failure is associated with a predefined response. The correct response depends on mission and environmental context. Compartmentalization constrains the failed

function, while the degraded-operation policy determines subsequent behavior. Quarantine without a tested recovery policy does not by itself guarantee safety. These operations policies are summarized in Table 2.

Table 2: Conceptual degraded-operation policies

Affected compartment	Response
Payload	Disable payload and preserve safe flight
Mission autonomy	Use the last validated plan or return home
Communication	Apply the predefined link-loss procedure
Navigation source	Switch to an independent source when available
Perception	Reduce autonomy, hold, return, or land
Data management	Preserve essential telemetry and suspend noncritical writes
Safety-critical service	Activate an independent fail-safe procedure

RESEARCH METHODOLOGY AND EVALUATION FRAMEWORK

This study follows a design-science methodology comprising system characterization, threat analysis, architectural decomposition, policy specification, scenario development, and comparative analytical evaluation [42].

System Characterization

A representative mission-critical UAV is characterized by its functions, trust levels, communication paths, timing requirements, resource dependencies, and safety consequences. This analysis determines which functions should be separated and which interactions are necessary.

The scope assumes a UAV with a flight-control unit, a companion-computing capability, one or more mission payloads, wireless command and telemetry interfaces, and autonomous navigation or perception functions. Highly resource-constrained UAVs are outside the primary scope.

Threat and Failure Analysis

For each threat or failure, the analysis identifies:

1. The originating compartment;
2. The affected asset or function;
3. Permitted propagation paths;
4. Potential safety and mission consequences;
5. Applicable detection or validation controls; and
6. The expected restriction, quarantine, or recovery action.

The examined events include communication compromise, malicious or faulty payload software, malformed or replayed messages, navigation spoofing, unsafe autonomy

output, unauthorized actuator access, resource exhaustion, process crashes, and communication loss. The analysis draws on established threat-modeling and hazard-analysis practice as referenced in [43] and [44].

Comparative Configurations

The analytical framework compares:

- **Conventional architecture:** components have broad internal communication and may share privileges or resources.
- **Compartmentalized architecture:** functions occupy separate trust domains, and cross-domain communication is explicitly mediated.

Equivalent functions, missions, threats, and failure assumptions are used in both configurations. This controls the comparison conceptually and focuses the analysis on the effect of isolation and mediation.

Evaluation Scenarios

Representative scenarios include:

- Payload-process failure;
- Malformed communication messages;
- Message flooding;
- Replay or delayed-message attacks;
- False navigation measurements;
- Unsafe waypoint recommendations;
- Unauthorized actuator-access attempts;
- Communication loss; and
- Processor, memory, storage, or bandwidth exhaustion.

The scenarios cover direct unauthorized access, data-integrity attacks, availability attacks, software failures, and loss of external services [9] [17].

Evaluation Metrics

The assessment considers fault-propagation scope, containment, expected detection and response, mission continuity, safe recovery, trajectory deviation, safety violations, and anticipated latency and resource overhead. Containment is expressed as

$$CR = 1 - \frac{N_{\text{affected}} - 1}{N_{\text{total}} - 1}, \quad (8)$$

where N_{affected} is the number of affected compartments (including the originating compartment) and N_{total} is the total number of compartments. CR ranges from 0 to 1. A value of 1 indicates that the event remains confined to its originating compartment, and a

value of 0 indicates that it has propagated to every compartment. For example, in the seven-compartment architecture of Section 4, a payload failure confined to the payload compartment yields $CR = 1$, whereas one that also reaches mission autonomy yields $CR = 1 - 1/6 \approx 0.83$.

Because this paper does not report a prototype or experiment, detection rate, latency, energy use, and related quantitative metrics are defined for future validation rather than reported as measured results.

ANALYTICAL EVALUATION AND DISCUSSION

The architecture is evaluated by tracing each event from its originating compartment through the authorized communication and resource dependencies. The analysis determines whether the event would be rejected, restricted, quarantined, or allowed to affect another compartment. Table 3 summarizes the comparison.

Table 3: Comparative analytical evaluation

Scenario	Conventional architecture	Compartmentalized response
Payload crash	May affect shared processes or resources	Confined to payload compartment
Malformed message	May reach internal services	Rejected through schema validation
Message flooding	May degrade multiple functions	Constrained by rate and resource limits
Replay attack	May cause stale control action	Rejected through freshness checks
Navigation spoofing	May influence route decisions	Restricted and cross-checked
Unsafe autonomy output	May become a control request	Rejected by safety validation
Actuator-access attempt	May exploit broad privileges	Blocked by the trust boundary
Communication loss	May interrupt mission control	Local fail-safe remains available
Resource exhaustion	May starve critical services	Constrained by compartment budgets

Containment of Exposed Components

The analysis suggests that compartmentalization is most effective for failures originating in communication, payload, mission, and perception functions. These domains are externally exposed or software-intensive but do not require direct actuator access. Restricting their interfaces removes pathways through which a compromise could directly affect flight control.

For example, a payload crash may cause loss of inspection capability but should not terminate flight-control services. Similarly, a compromised communication service may submit only message types explicitly authorized by policy. The gateway rejects unauthorized types, invalid values, stale messages, and excessive rates before they reach higher-criticality compartments.

The benefit is therefore not uninterrupted mission execution in every case. Instead, the architecture converts some system-level failures into localized mission degradations or safe aborts.

Safety of Autonomous Decisions

Mission-planning and artificial-intelligence components can produce unsafe outputs because of compromise, model error, unexpected conditions, or incorrect inputs. Treating their outputs as recommendations prevents them from exercising direct actuator authority. Geofence, altitude, speed, energy, and flight-mode checks provide an independent safety layer. This mechanism is limited by the completeness of the safety model. A command may satisfy all specified constraints and still be inappropriate in an unforeseen context. Compartmentalization therefore reduces authority but cannot guarantee the correctness of autonomous decisions.

Navigation and Sensor Limitations

Navigation spoofing illustrates the distinction between containment and truth determination. Isolation can prevent one navigation source from directly modifying unrelated functions, and trust management can reduce the authority of an inconsistent source. However, the architecture cannot determine that plausible sensor data are false without independent references, physical models, or redundant sensing [45].

Accordingly, compartmentalization must be combined with sensor diversity, consistency checking, and resilient state estimation [46]. If no reliable alternative exists, the safe response may be to reduce autonomy, hold position, return home, or land.

Resource Isolation

Memory and interface separation alone are insufficient. A malicious or faulty component could interfere with flight-critical operation by consuming processor time, memory, storage, network bandwidth, or power. The proposed model therefore includes per-compartment resource constraints and prioritization of critical traffic.

Resource controls must be selected carefully. Limits that are too high permit interference, while limits that are too low may disrupt legitimate peak workloads. Actual budgets require experimental characterization of the target platform and mission.

Trust and Response Trade-Offs

Deterministic violations, such as malformed messages or unauthorized actuator access, support immediate rejection. Behavioral anomalies are less certain and may require accumulated evidence. The normal, restricted, and quarantined states allow a graduated response to uncertainty. False positives remain an important risk. Incorrectly restricting a nonessential payload may only reduce mission capability, whereas incorrectly quarantining navigation can trigger an emergency procedure. Thresholds should therefore account for component criticality and the consequences of both delayed and unnecessary intervention.

Security and Performance Trade-Off

The expected costs of compartmentalization include cross-domain communication latency, processing and memory overhead, energy use, policy-management complexity, and additional assurance requirements. Energy is already a binding constraint for many UAVs. When it is not properly budgeted and monitored, designers must make trade-offs that can compromise the UAV's intended use [24], [25]. Any additional overhead competes for the same limited budget. These costs make full compartmentalization more appropriate for medium-to-large mission-critical UAVs than for resource-constrained platforms. A platform remains suitable only if the additional latency preserves its timing requirement:

$$L_{\text{base}} + L_{\text{mediation}} < D_{\text{max}}, \quad (9)$$

where L_{base} is the existing processing latency, $L_{\text{mediation}}$ is the added policy and validation delay, and D_{max} is the maximum allowable deadline.

The architecture should therefore be tailored rather than applied uniformly. A smaller UAV may isolate only flight control, communication, and payload processing, while a higher-assurance platform may use all seven compartments.

Broader Operational Challenges

UAVs are classified by configuration, weight, endurance, cost, size, and power consumption, and each class faces its own operational challenges. Several of these challenges lie outside what compartmentalization can solve, but they shape how the architecture must respond.

Weather

Adverse weather can cause a UAV to deviate from its plan or be lost entirely. Recreational operators may lose control of their drones, resulting in damage, loss, and customer dissatisfaction [9]. In agriculture, wind and other weather conditions can cause spray drift and uneven deposition, wasting costly materials such as pesticides [54]. High winds, heavy rain, snow, or storms can force delivery UAVs off course, delaying delivery or causing loss of the payload. Lightning and high winds can delay emergency response, with serious consequences for the people, property, or areas involved [9]. Weather can also degrade UAV-provided network coverage by forcing UAVs out of the positions planned for optimal coverage [22].

Compartmentalization cannot mitigate these physical hazards. However, weather is a common non-adversarial cause of the navigation degradation and communication loss examined in Sections 6.3 and 4.6. The same degraded-operation policies, such as holding position, returning home, or landing, therefore apply whether the cause is an attack or the environment.

Security and Privacy

Like any system that depends on data acquisition and transfer, UAVs face serious security risks. Breaches can cause loss of control, physical loss of the UAV, or exposure of sensitive mission data. Figure 4 illustrates a UAV network and its potential points of attack. The

required level of security and privacy varies by application [19]. Being able to swap or adapt software and hardware components to match an application's needs can reduce cost and variation while preserving the required protection. This supports the tailored deployment described in Section 6.6.

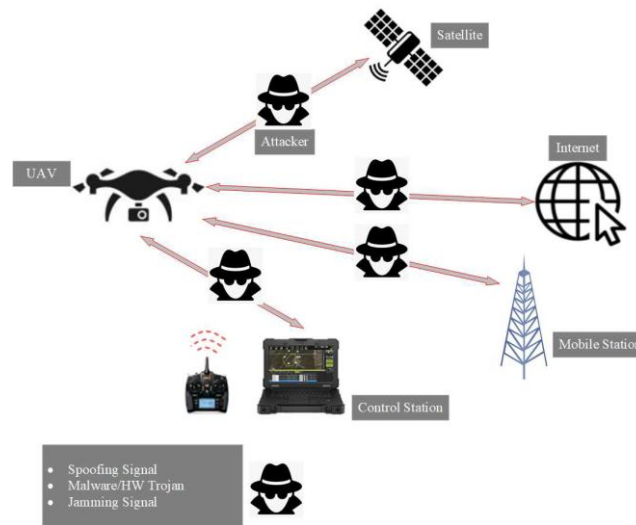


Figure 4: UAV Network and potential areas of threat [13]

Payload Safety

Most UAVs have low payload capacity because of physical and technical limitations. An unintended payload release can damage property. In agriculture, it can waste the materials needed to maintain a crop, or destroy the crop if the entire load is released at once. In the worst case, a payload dropped on a person or into a congested area can cause serious injury or death [23]. FAA registration requirements begin at a takeoff weight of 250 g [20], [21], but registration alone does not prevent such accidents. Even a payload of a few kilograms falling from operating altitude can strike with enough energy to cause serious injury. The risk increases when a UAV is used for an application it was not designed for. Within the proposed architecture, payload release is a physical action with safety consequences. It should therefore be treated as a flight-related request subject to the independent safety validation of Section 4.4, not as an unrestricted payload function.

Platform Variation

Hardware, software, and physical limitations restrict most UAVs to a single task, or one task at a time. Existing variation is largely physical: fixed-wing, multirotor, single-rotor, and hybrid fixed-wing designs, each suited to particular uses [26]. When a field requires multiple tasks and each task requires a different type of UAV, costs rise. A compartmentalized architecture with modular payload compartments could allow one platform to support multiple mission roles more safely, because adding or replacing a payload would not expand its authority over flight-critical functions.

Overall Assessment

The analytical comparison indicates four principal advantages:

- Reduced direct exposure of flight-critical functions;
- Smaller fault- and attack-propagation paths;
- Explicit control over trust and cross-domain authority; and
- Preservation of safe degraded operation after some noncritical failures.

The principal limitations are dependence on the trusted computing base, incomplete knowledge of sensor truth, added resource costs, the possibility of incorrect policy or trust thresholds, and the inability to mitigate physical hazards such as weather. The proposed architecture should therefore be viewed as one layer of defense in depth rather than a complete security solution.

LIMITATIONS AND FUTURE WORK

Limitations

This study proposes and evaluates the architecture conceptually; it does not report implementation or flight-test results. Consequently, quantitative claims about latency, detection accuracy, energy consumption, and mission success remain for future validation.

The model also assumes that the flight controller, isolation mechanism, policy core, and safety monitor remain trustworthy. A compromise of these components could bypass or disable containment. Furthermore, results may vary across UAV sizes, autopilots, processors, operating systems, payloads, and missions.

Experimental Validation

Future work should evaluate the architecture through software-in-the-loop, hardware-in-the-loop, and controlled flight testing. Experiments should measure timing, processing, memory, and energy overhead while injecting single and coordinated failures. Additional work should compare isolation mechanisms, formally verify communication policies, investigate redundant safety monitors, and extend compartmentalization to multi-UAV trust domains.

Toward a Modular, Compartmentalized UAV Platform

A further direction addresses the platform-variation challenge identified in Section 6.7.4. Existing UAV designs [26] each have limitations in area access, takeoff and landing, robustness, and other factors. As a result, organizations may need to buy several UAV types to complete one mission or meet their operational needs, which is costly and time-consuming. Table 4 summarizes the challenges from Section 6.7 and how a modular, compartmentalized platform could address them.

Earlier efforts have customized the UAV for each individual application, which can be costly and inefficient [18]. A common platform with interchangeable components would reduce the need for that customization. The automotive industry offers a precedent in the Chrysler K platform, used from 1981 to 1989 [27]. It was intended to standardize parts and

reduce variation, yet it also made customization easy and affordable: the platform could be shortened, stretched, or reconfigured as needed [28].

Table 4: UAV challenges and potential modular-platform approaches

Challenge	Limitation of fixed designs	Modular, compartmentalized approach
Weather	A fixed airframe may lack robustness for expected conditions	Select chassis and aerodynamic options suited to the operating environment
Power and energy	A fixed battery forces trade-offs against the mission	Interchangeable battery modules by range class
Payload	Low capacity; unsafe release can cause harm	Chassis class matched to payload weight; payload release validated by the safety compartment
Security and privacy	Uniform security may be excessive or insufficient	Communication modules and compartment sets matched to required assurance
Platform variation	Single-task designs require multiple fleets	Common platform with interchangeable subsystems supports multiple roles

Figure 5 shows a proposed UAV subsystem strategy. The battery, controller, network/GPS, and I/O and sensor subsystems each offer multiple interchangeable options matched to the intended application. The chassis and aerodynamic subsystems are drawn from a smaller set of options designed to serve multiple applications and to be compatible with all other subsystems.

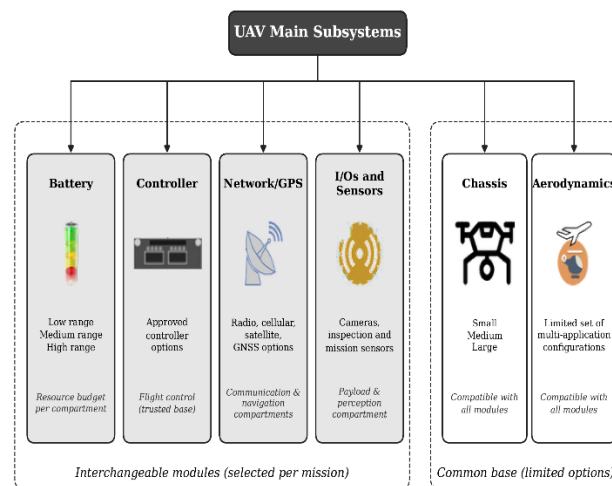


Figure 5: Proposed modular UAV Subsystems

The software compartmentalization proposed in this paper is what would make such interchange safe. Each interchangeable module would enter through a defined compartment with fixed permissions, so changing the hardware configuration would not change the trust boundary around flight control. The mission would determine which modules are installed, and the compartment policies would determine what those modules are allowed to do. One exception follows from Section 7.1: an interchangeable flight-controller module would form part of the trusted computing base, so each controller option would require its own assurance.

For example, a UAV configured for agriculture might use a chassis from a predefined size class (small, medium, or large) chosen for its payload requirements, and a battery from a range class (low, medium, or high) chosen for field size. The remaining subsystems would be chosen according to the required sensing and application tasks. Defining these selection criteria, the interfaces between modules, and the compartment policies for each module type is left for future research.

CONCLUSION

This paper reviewed major UAV applications and the challenges that affect them, including weather, energy limitations, payload safety, security and privacy, and platform variation. In response, it proposed a compartmentalized resilience architecture for autonomous, network-connected, mission-critical UAVs. The architecture separates flight control, navigation, communication, mission autonomy, payload processing, data management, and safety monitoring into distinct trust domains. Default-deny mediation, independent safety validation, adaptive trust, quarantine, and predefined degraded modes limit the influence of failed or compromised components.

The approach is most applicable to software-rich UAVs with exposed interfaces, modular payloads, sensitive data, and sufficient onboard resources. It is not intended as a universal design for every UAV class. The analytical evaluation suggests that compartmentalization can reduce propagation paths and preserve safe operation, especially when failures originate in communication, payload, mission, or perception functions. Experimental validation is required to determine whether these benefits outweigh real-time, energy, and resource costs on specific platforms.

Beyond resilience, compartmentalization provides a foundation for a modular UAV platform. Interchangeable battery, controller, network, and sensor modules would be built on a common chassis and aerodynamic base, an approach similar to the common-platform strategy Chrysler used in the 1980s. Because each module would interact with the system only through a defined compartment, a single platform could be reconfigured for different missions without expanding any module's authority over flight-critical functions. Such a platform could reduce the cost and time needed to deploy UAVs across applications and make the technology more accessible.

REFERENCES

- [1]. <https://interestingengineering.com/innovation/a-brief-history-of-drones-the-remote-controlled-unmanned-aerial-vehicles-uavs> as of July 23, 2023
- [2]. <https://www.iwm.org.uk/history/a-brief-history-of-drones> as of July 23, 2023
- [3]. <https://www.iwm.org.uk/history/a-brief-history-of-drones#:~:text=The%20first%20pilotless%20vehicles%20were,first%20flew%20in%20October%201918.> as of July 23, 2023
- [4]. <https://consortiq.com/uas-resources/short-history-unmanned-aerial-vehicles-uavs#:~:text=1935%20%E2%80%93%20The%20First%20Modern%20Drone%20is%20Developed&text=In%20response%2C%20the%20De%20Havilland,be%20the%20first%20modern%20drone.> as of July 23, 2023

- [5]. ResearchAndMarkets, "Drone Package Delivery Business Report 2025," Business Wire, Mar. 6, 2025. [Online]. Available: <https://www.businesswire.com/news/home/20250306610208/en/> (accessed Oct. 6, 2026)
- [6]. A. Alsalem and M. Zohdy, "A Review on Civil Applications of Vertical Take-Off and Landing Vehicles," 2023 IEEE Conference on Technologies for Sustainability (Sutekh), Portland, OR, USA, 2023, pp. 130-137, doi: 10.1109/SusTech57309.2023.10129574.
- [7]. <https://www.faa.gov/faq/what-definition-recreational-or-hobby-use-uas-or-drone#:~:text=Recreational%20or%20hobby%20UAS%20or,or%20for%20compensation%20or%20hire.>
- [8]. Federal Aviation Administration, "Drones by the Numbers." [Online]. Available: https://www.faa.gov/uas/resources/by_the_numbers (accessed Oct. 6, 2026)
- [9]. H. Shakhathreh et al., "Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges," in IEEE Access, vol. 7, pp. 48572-48634, 2019, doi: 10.1109/ACCESS.2019.2909530.
- [10]. H. S. Munawar, H. Inam, F. Ullah, S. Qayyum, A. Z. Kouzani, and M. A. P. Mahmud, "Towards Smart Healthcare: UAV-Based Optimized Path Planning for Delivering COVID-19 Self-Testing Kits Using Cutting Edge Technologies," Sustainability, vol. 13, no. 18, p. 10426, Sep. 2021, doi: 10.3390/su131810426.
- [11]. UAV and obstacle sensing techniques - a perspective; Aswini, N; Kumar, E Krishna; Uma, S V. International Journal of Intelligent Unmanned Systems; Bingley Vol. 6, Iss. 1, (2018): 32-46. DOI:10.1108/IJIUS-11-2017-0013
- [12]. N. Muchiri and S. Kimathi, "A review of applications and potential applications of UAV," in Proc. Sustain. Res. Innov. Conf., 2016
- [13]. Elmeseiry, Nourhan & Alshaer, Nancy & Ismail, Tawfik. (2021). A Detailed Survey and Future Directions of Unmanned Aerial Vehicles (UAVs) with Potential Applications. Aerospace Engineering. 10.3390/aerospace8120363.
- [14]. <https://www.farmbrite.com/post/how-to-prepare-a-field-for-planting> as of July 23, 2023
- [15]. J. Kim, S. Kim, C. Ju and H. I. Son, "Unmanned Aerial Vehicles in Agriculture: A Review of Perspective of Platform, Control, and Applications," in IEEE Access, vol. 7, pp. 105100-105115, 2019, doi: 10.1109/ACCESS.2019.2932119
- [16]. L. Davies, Y. Vagapov, V. Grout, S. Cunningham and A. Anuchin, "Review of Air Traffic Management Systems for UAV Integration into Urban Airspace," 2021 28th International Workshop on Electric Drives: Improving Reliability of Electric Drives (IWED), 2021, pp. 1-6, doi: 10.1109/IWED52055.2021.9376343.
- [17]. Y. Zeng, R. Zhang and T. J. Lim, "Wireless communications with unmanned aerial vehicles: opportunities and challenges," in IEEE Communications Magazine, vol. 54, no. 5, pp. 36-42, May 2016, doi: 10.1109/MCOM.2016.7470933.
- [18]. Singhal, G.; Bansod, B.; Mathew, L. Unmanned Aerial Vehicle Classification, Applications and Challenges: A Review. Preprints 2018, 2018110601. <https://doi.org/10.20944/preprints201811.0601.v1>
- [19]. Menouar, H., Guvenc, I., Akkaya, K., Uluagac, A.S., Kadri, A. and Tuncer, A., 2017. UAV-enabled intelligent transportation systems for the smart city: Applications and challenges. IEEE Communications Magazine, 55(3), pp.22-28.
- [20]. <https://www.faa.gov/uas> as of July 31, 2023
- [21]. <https://dronesurveyservices.com/drone-statistics/> as of July 31, 2023

- [22]. Shrestha, Rakesh, Inseon Oh, and Shiho Kim. "A survey on operation concept, advancements, and challenging issues of urban air traffic management." *Frontiers in Future Transportation* 2 (2021): 1.
- [23]. Mohsan, S.A.H., Othman, N.Q.H., Li, Y. et al. Unmanned aerial vehicles (UAVs): practical aspects, applications, open challenges, security issues, and future trends. *Intel Serv Robotics* 16, 109-137 (2023). <https://doi.org/10.1007/s11370-022-00452-4>
- [24]. L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in UAV communication networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1123-1152, 2nd Quart., 2016.
- [25]. B. Vergouw, H. Nagel, G. Bondt, and B. Custers, "Drone technology: Types, payloads, applications, frequency spectrum issues and future developments," in *The Future of Drone Use*. The Hague, Netherlands: TMC Asser Press, 2016, pp. 21-45.
- [26]. <https://www.auav.com.au/articles/drone-types/> as of July 23rd, 2023
- [27]. https://en.wikipedia.org/wiki/Chrysler_K_platform as of July 23, 2023
- [28]. <https://www.automotivehalloffame.org/quality-with-a-k/#:~:text=The%20genius%20of%20the%20K,any%20shape%20the%20company%20desired.> As of July 23, 2023
- [29]. <https://www.politico.com/story/2013/12/obstacles-to-the-jeff-bezos-drone-dream-100536> as of July 31, 2023
- [30]. <https://web.archive.org/web/20140714152133/http://www.seattlebulletin.com/index.php/sid/223727243/scat/9d7afd9766a94f28/ht/E-commerce-giant-Amazon-seeks-FAA-nod-for-testing-drones> as of July 31, 2023
- [31]. <https://artsandculture.google.com/story/america%E2%80%99s-first-commercial-drone-delivery-service-smithsonian-national-air-and-space-museum/8wXx3rcHpj8VLQ?hl=en> as of July 31, 2023
- [32]. <https://www.theverge.com/2022/4/4/23006894/alphabet-wing-drone-delivery-dallas-kick-off> as of July 31, 2023
- [33]. <https://wing.com/how-it-works/> as of July 23, 2023
- [34]. <https://www.supplychaindive.com/news/innovator-ups-flight-forward-drone-delivery-dive-awards/566093/> as of July 23, 2023
- [35]. <https://www.welt.de/wirtschaft/article122747484/DHL-testet-erstmal-Paketlieferung-per-Drohne.html> as of July 23, 2023
- [36]. <https://www.freightwaves.com/news/dhl-pulling-its-parcelcopter-drone-ceasing-drone-development#:~:text=International%20shipping%20giant%20DHL%20is,in%20the%20emerging%20drone%20industry.> as of July 23, 2023
- [37]. Ahmad H. Sawalmeh, Noor Shamsiah Othman; An Overview of Collision Avoidance Approaches and Network Architecture of Unmanned Aerial Vehicles (UAVs), <https://doi.org/10.48550/arXiv.2103.14497>
- [38]. H. A. Hashim, "Advances in UAV avionics systems architecture, classification and integration: A comprehensive review and future perspectives," *Results in Engineering*, vol. 25, Art. no. 103786, 2025, doi: 10.1016/j.rineng.2024.103786.
- [39]. J. Vanlyssel et al., "Trust without boundaries: An architectural analysis of satellite flight software," *arXiv:2608.14532*, 2026.

- [40]. C. H. Kim et al., "Securing real-time microcontroller systems through customized memory view switching," in Proc. 25th Annu. Netw. Distrib. Syst. Secur. Symp. (NDSS), San Diego, CA, USA, 2018, pp. 1-15.
- [41]. A. Khan, D. Xu, and D. J. Tian, "Low-cost privilege separation with compile time compartmentalization for embedded systems," in 2023 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2023, pp. 2577-2594.
- [42]. K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45-77, 2007.
- [43]. A. Shostack, *Threat Modeling: Designing for Security*. Indianapolis, IN, USA: John Wiley & Sons, 2014.
- [44]. N. G. Leveson, *Engineering a Safer World: Systems Thinking Applied to Safety*. Cambridge, MA, USA: MIT Press, 2011.
- [45]. A. J. Kerns, D. P. Shepard, J. A. Bhatti, and T. E. Humphreys, "Unmanned aircraft capture and control via GPS spoofing," *Journal of Field Robotics*, vol. 31, no. 4, pp. 617-636, 2014, doi: 10.1002/rob.21513.
- [46]. M. L. Psiaki and T. E. Humphreys, "GNSS spoofing and detection," *Proceedings of the IEEE*, vol. 104, no. 6, pp. 1258-1270, 2016, doi: 10.1109/JPROC.2016.2526658.
- [47]. Y. Mekdad, A. Aris, L. Babun, A. El Fergougui, M. Conti, R. Lazzeretti, and A. S. Uluagac, "A survey on security and privacy issues of UAVs," *Computer Networks*, vol. 224, Art. no. 109626, 2023, doi: 10.1016/j.comnet.2023.109626.
- [48]. J. Rushby, "Design and verification of secure systems," in Proc. 8th ACM Symp. Operating Syst. Principles (SOSP), Pacific Grove, CA, USA, 1981, pp. 12-21.
- [49]. G. Klein, J. Andronick, M. Fernandez, I. Kuz, T. Murray, and G. Heiser, "Formally verified software in the real world," *Communications of the ACM*, vol. 61, no. 10, pp. 68-77, 2018, doi: 10.1145/3230627.
- [50]. H.-C. Jo, J.-K. Park, H.-W. Jin, H.-S. Yoon, and S. H. Lee, "Portable and configurable implementation of ARINC-653 temporal partitioning for small civilian UAVs," *IEEE Access*, vol. 7, pp. 142478-142487, 2019, doi: 10.1109/ACCESS.2019.2944662.
- [51]. A. Farrukh and R. West, "FlyOS: Rethinking integrated modular avionics for autonomous multicopters," *Real-Time Systems*, vol. 59, no. 2, pp. 256-301, 2023, doi: 10.1007/s11241-023-09399-w.
- [52]. A. Clements, N. S. Almakhdhub, S. Bagchi, and M. Payer, "ACES: Automatic compartments for embedded systems," in Proc. 27th USENIX Security Symp. (USENIX Security), Baltimore, MD, USA, 2018, pp. 65-82.
- [53]. L. Sha, "Using simplicity to control complexity," *IEEE Software*, vol. 18, no. 4, pp. 20-28, 2001, doi: 10.1109/MS.2001.936213.
- [54]. W. Qin and P. Chen, "Analysis of the research progress on the deposition and drift of spray droplets by plant protection UAVs," *Scientific Reports*, vol. 13, Art. no. 14935, 2023, doi: 10.1038/s41598-023-40556-0.