

Dissertation/Report submitted to the
UNIVERSITY OF MOHAMED BOUDIAF - M'SILA, ALGERIA



FACULTY OF MATHEMATICS AND COMPUTER SCIENCE
DEPARTMENT OF COMPUTER SCIENCE

in partial fulfillment of the requirements for the degree of

Master's in Computer Science

Specialization: Computer Science

Option: Networks and Information and Communication Technologies

Presented by

Imane Aid

Entitled

Exhaustive Distributed Intrusion Detection System for UAVs Attacks Detection

Under the supervision of

Lamri Sayad

Jury Members

Dr. Rached Yagoubi

University of M'sila

President

Prof. Lamri Sayad

University of M'sila

Supervisor

Dr. Abdelouahab Khettaf

University of M'sila

Examiner

June 7th, 2026

Dedication

“Other things may change us, but we start and end with the family.” — Anthony Brandt

This work is dedicated to the people who supported me throughout this journey.

To my dad, who taught me strength, discipline, and how to keep moving forward even when things get difficult.

To my mom, who stood by me with patience and care.

And to myself, for the effort, persistence, and dedication invested in completing this work.

لله الحمد

Imane

Acknowledgements

I would like to express my sincere gratitude to my supervisor **Prof.Lamri Sayad** for their guidance, support, and valuable feedback throughout this work. Their expertise and constructive advice were essential in shaping the direction of this thesis and improving its quality.

I also extend my thanks to everyone who contributed, to the completion of this research.

Contents

Introduction	1
1 Chapter 1: Literature Review	3
1.1 Introduction	3
1.2 Unmanned Aerial Vehicles (UAVs):	3
1.2.1 Definition of UAV:	3
1.2.2 Unmanned Aerial System (UAS)	4
1.2.3 Types and Characteristics of UAVs	5
1.2.4 Design Considerations of UAVs	6
1.3 Flying Ad-Hoc Networks (FANETs)	7
1.3.1 FANET Characteristics	7
1.4 UAV Communication	9
1.4.1 UAV Communication Architectures	9
1.5 Taxonomy of UAV Attack Types	11
1.5.1 Resource Exhaustion and Availability Attacks	11
1.5.2 Communication Channel Attacks	12
1.5.3 Mobility and Navigation Attacks	12
1.5.4 Network Topology and Routing Attacks	13
1.5.5 Application-Level and Payload Attacks	13
1.6 Intrusion Detection Systems (IDS)	14
1.6.1 Definition and Goal of an IDS	14
1.6.2 IDS Classification	14
1.6.3 IDS Deployment Strategies	17
1.7 Kolmogorov–Arnold Networks (KAN)	18
1.7.1 Kolmogorov–Arnold Representation Theorem	19
1.7.2 From Theorem to KAN Architecture	19
1.7.3 KAN vs MLP	20
1.7.4 Why KAN for UAV Intrusion Detection	21
1.8 Conclusion	21
2 Chapter 2: Related Work	22

2.1	Introduction	22
2.2	Different intrusion detection systems for UAVs security	22
2.2.1	Data Collection Paradigms	23
2.2.2	Deployment Architectures	24
2.2.3	Optimization Strategies	25
2.2.4	Detection Methodologies	25
2.2.5	Discussion	26
2.3	Reference Architecture : E-DIDS	28
2.3.1	Overview of E-DIDS	28
2.3.2	E-DIS Proposed Scheme	28
2.3.3	System model and proposed scheme	31
2.4	Conclusion	33
3	Chapter 3: Methodology	34
3.1	Introduction	34
3.2	Adopted Architecture	35
3.3	System Modules	36
3.3.1	Sensor Module	37
3.3.2	Communication Module	37
3.3.3	Management Module	38
3.3.4	Central Control Module	40
3.3.5	Reporting Module	42
3.4	Fleet Organization	43
3.5	Cluster-Based Organization	43
3.5.1	Cluster Head	44
3.5.2	Cluster Head Election	45
3.6	Data Collection and Preprocessing	46
3.6.1	Datasets Description	46
3.6.2	Preprocessing Pipeline	48
3.7	Kolmogorov-Arnold Network (KAN) Classifier	50
3.7.1	Adaptive Grid Construction	50
3.7.2	Architectural Enhancements	51
3.7.3	Classification Objective	51
3.7.4	Inference	51
3.7.5	Summary	52
3.8	Intrusion Detection Framework	52
3.9	Intrusion Detection Subsystems	52
3.9.1	Network Intrusion Detection System	52

3.9.2	Navigation and Telemetry Intrusion Detection System	53
3.9.3	Embedded Communication Intrusion Detection System	54
3.9.4	Intrusion Detection Engine (Runtime Inference Layer)	55
3.10	Overall System Workflow	57
3.11	Evaluation Metrics	59
3.11.1	Classification Performance Metrics	59
3.11.2	Operational Performance Metrics	60
3.12	Conclusion	61
4	Chapter 4: System Implementation and Performance Evaluation	62
4.1	Introduction	62
4.2	Implementation	62
4.2.1	System Modules	63
4.2.2	Cluster Head Election Protocol	65
4.2.3	Preprocessing	66
4.2.4	Intrusion Detection Systems	68
4.3	Results Discussion	72
4.3.1	Network Intrusion Detection System Results	72
4.3.2	Navigation and Telemetry Intrusion Detection System Results . . .	74
4.3.3	Embedded Communication Intrusion Detection System (UAVCAN)	75
4.4	Limitations of the Work	77
4.5	Conclusion	78
	Conclusion	79

List of Figures

1.1	Unmanned Aerial Vehicle architecture.	4
1.2	Components of a typical Unmanned Aerial System	5
1.3	Types of Unmanned Aerial System	6
1.4	Extend coverage area by using FANET.	8
1.5	UAV Communication Architecture Taxonomy.	9
1.6	Centralised UAV Communication Architectures.	10
1.7	Decentralised UAV communication architectures.	10
1.8	UAV Attack Taxonomy.	11
1.9	Intrusion Detection Systems Classification	15
1.10	Notations of activations that flow through the network.	20
1.11	Multi-Layer Perceptrons (MLPs) vs. Kolmogorov-Arnold Networks (KANs).	20
2.1	Intrusion Detection Systems (IDS) Taxonomy in UAV Networks	23
2.2	Diagram scheme of the E-DIDS modules.	29
2.3	An illustration of functional attacks detection in a mapped representation of UAV flight log	31
2.4	High-level flow diagram of the E-DIDS execution.	33
3.1	Overall System Architecture.	35
3.2	Cluster Head Election Protocol	46
3.3	Data Preprocessing Pipeline and KAN Model Architecture	52
3.4	System Workflow.	59
4.1	Confusion matrix for Network IDS (Blackhole, Flooding, Sybil, Wormhole, Normal).	73
4.2	Confusion matrix for MAVLink IDS (GPS Jamming, GPS Spoofing, Normal).	75
4.3	Confusion matrix for UAVCAN IDS (Flooding, Fuzzy, Normal).	77

List of Tables

2.1	Comparative summary of representative intrusion detection systems for UAV security.	27
3.1	Comparison between the original E-DIDS architecture and the proposed implementation.	36
3.2	Summary of intrusion detection subsystems used in the proposed E-DIDS architecture.	55
4.1	Overall performance comparison of the proposed E-DIDS architecture across UAV communication layers.	72
4.2	Overall performance of the Network IDS on the test set.	72
4.3	Per-class performance of the Network IDS.	73
4.4	Overall performance of the MAVLink IDS.	74
4.5	Per-class performance of MAVLink IDS.	74
4.6	UAVCAN IDS training progression	76
4.7	UAVCAN IDS test metrics	76

Abstract

Unmanned Aerial Vehicles (UAVs) are increasingly deployed in modern intelligent systems due to their ability to operate autonomously and collaboratively in complex environments. However, this connectivity introduces significant cybersecurity risks, as communication channels and control exchanges become potential targets for malicious attacks.

This thesis proposes a distributed intrusion detection framework tailored for UAV networks, where each UAV performs local detection while contributing to a global collaborative decision-making process. Kolmogorov-Arnold Networks (KANs) are adopted as the main learning model due to their ability to efficiently capture nonlinear patterns with low computational cost, making them suitable for resource-constrained UAV environments.

Three intrusion detection scenarios are studied: network traffic monitoring, MAVLink communication security, and UAVCAN communication security. Experimental results show that the proposed approach achieves strong detection performance while maintaining scalability and adaptability in dynamic UAV environments.

Keywords: UAV, Intrusion Detection System, Machine Learning, Kolmogorov-Arnold Networks, Network Security.

Résumé

Les véhicules aériens sans pilote (UAV) sont de plus en plus utilisés dans les systèmes intelligents modernes grâce à leur capacité à fonctionner de manière autonome et collaborative dans des environnements complexes. Cependant, cette connectivité introduit des risques importants en matière de cybersécurité, les canaux de communication devenant des cibles potentielles d'attaques.

Ce mémoire propose une architecture distribuée de détection d'intrusion adaptée aux réseaux UAV, où chaque drone effectue une détection locale tout en participant à un processus de décision global collaboratif. Les réseaux de Kolmogorov-Arnold (KAN) sont utilisés comme modèle principal en raison de leur efficacité dans la modélisation des relations non linéaires avec un coût de calcul réduit.

Trois scénarios sont étudiés : la surveillance du trafic réseau, la sécurité des communications MAVLink et UAVCAN. Les résultats expérimentaux montrent de bonnes performances avec une bonne capacité d'adaptation et de passage à l'échelle.

Mots-clés : UAV, détection d'intrusion, apprentissage automatique, réseaux de Kolmogorov-

الملخص

تُستخدم الطائرات بدون طيار (UAV) بشكل متزايد في الأنظمة الذكية الحديثة بفضل قدرتها على العمل بشكل مستقل وتعاوني في بيئات معقدة. ومع ذلك، فإن هذا الترابط يخلق تهديدات أمنية سيبرانية جديدة، حيث تصبح قنوات الاتصال عرضة للهجمات. تقترح هذه الأطروحة إطاراً موزعاً لكشف التسلل مخصصاً لشبكات الطائرات بدون طيار، بحيث تقوم كل طائرة بتنفيذ كشف محلي مع المساهمة في عملية اتخاذ قرار جماعي على مستوى النظام. يتم اعتماد شبكات Kolmogorov-Arnold (KAN) كنموذج رئيسي بسبب قدرتها على تمثيل العلاقات غير الخطية بكفاءة مع تكلفة حسابية منخفضة. تتم دراسة ثلاثة سيناريوهات: مراقبة حركة الشبكة، تأمين اتصالات MAVLink، وتأمين اتصالات UAVCAN. تُظهر النتائج التجريبية أداءً جيداً مع قابلية عالية للتوسع والتكيف في بيئات UAV الديناميكية.

الكلمات المفتاحية: الطائرات بدون طيار، كشف التسلل، التعلم الآلي، شبكات Kolmogorov-Arnold، أمن الشبكات.

Introduction

The development of Unmanned Aerial Vehicles (UAVs) has progressed rapidly over recent years. UAVs are no longer simple remotely controlled machines; they have evolved into intelligent autonomous systems capable of performing complex missions. Modern UAVs are able to communicate with each other and operate collaboratively, which allows them to accomplish tasks with minimal human intervention.

Today, UAVs are widely used in several domains such as environmental monitoring, precision agriculture, surveillance, disaster management, infrastructure inspection, and military applications. Their ability to operate individually or as part of a coordinated swarm has significantly expanded their potential and flexibility.

As UAV technology continues to advance, these systems increasingly rely on communication networks and collaborative decision-making mechanisms to achieve their objectives. While this connectivity improves efficiency and coordination among multiple UAVs, it also introduces new security risks. Communication links, control messages, and exchanged data may become targets for cyberattacks that aim to disrupt operations, manipulate information, or compromise mission integrity. In critical environments, such attacks can lead to severe consequences, making security a fundamental requirement rather than an optional feature.

Among the available security mechanisms, Intrusion Detection Systems (IDSs) play an essential role in identifying malicious activities and abnormal behaviors within a system. However, designing IDS solutions for UAV environments remains a challenging task. UAV networks are highly dynamic, resource-constrained, and operate under constantly changing conditions. As a result, traditional intrusion detection approaches designed for conventional networks are often not sufficient for such environments.

Recent advances in machine learning have opened new opportunities to address these challenges. Machine learning models are capable of learning complex patterns from data and adapting to evolving attack strategies. In this work, particular attention is given to Kolmogorov-Arnold Networks (KANs), a recent neural network architecture that provides strong nonlinear modeling capabilities while maintaining computational efficiency. These characteristics make KANs particularly suitable for UAV-based intrusion detection systems.

The objective of this thesis is to design and implement a distributed intrusion detection framework capable of securing UAV communications across multiple domains. The pro-

posed approach combines local detection at each UAV with collaborative decision-making mechanisms, allowing each node to independently identify suspicious activities while contributing to a global security view of the system. To evaluate the effectiveness of the proposed framework, three intrusion detection scenarios are considered: network traffic monitoring, MAVLink communication security, and UAVCAN communication security.

For better clarity, this thesis is organized into six main sections:

- **General Introduction:** Presents the context of the study, highlights the importance of UAV security, introduces the research problem, and outlines the objectives of the thesis.
- **Background and Theoretical Foundations:** Covers the fundamental concepts required to understand this work, including UAV systems, communication protocols, cybersecurity threats, machine learning techniques, and intrusion detection systems.
- **State of the Art:** Reviews existing research in UAV security and intrusion detection, analyzes current approaches, and identifies their limitations and research gaps.
- **Methodology and System Design:** Describes the proposed distributed intrusion detection architecture, system workflow, and the design of the KAN-based detection models.
- **Implementation and Experimental Evaluation:** Presents implementation details, datasets used, training process, and experimental results for the different scenarios.
- **Conclusion and Future Work:** Summarizes the main contributions of the thesis and suggests possible directions for future improvement.

This work aims to contribute to the development of more secure and resilient UAV networks by exploring distributed intrusion detection strategies and leveraging modern machine learning techniques to address emerging cybersecurity threats.

Chapter 1

Chapter 1: Literature Review

1.1 Introduction

This chapter presents the theoretical foundations of the study, focusing on UAV systems and their security challenges. It introduces Unmanned Aerial Vehicles (UAVs), Flying Ad-Hoc Networks (FANETs), and UAV communication architectures, highlighting their key characteristics and operational constraints.

It then discusses the main cyber threats targeting UAV environments, followed by a structured overview of Intrusion Detection Systems (IDS), including their classification and deployment strategies in distributed networks. Finally, Kolmogorov–Arnold Networks (KANs) are introduced as the learning paradigm motivating the proposed detection approach.

These concepts collectively establish the basis for the design and development of the proposed intrusion detection system.

1.2 Unmanned Aerial Vehicles (UAVs):

1.2.1 Definition of UAV:

An Unmanned Aerial Vehicle (UAV) is commonly considered to be a drone or an aircraft with no pilot on board. UAVs can be remote-controlled aircraft or they may operate autonomously along pre-programmed flight paths. The unmanned aerial vehicle contains cameras, sensors, communication modules, as well as other payload devices, enabling it to perform a wide variety of missions including surveillance, reconnaissance, delivery, and search-and-rescue operations.[1]

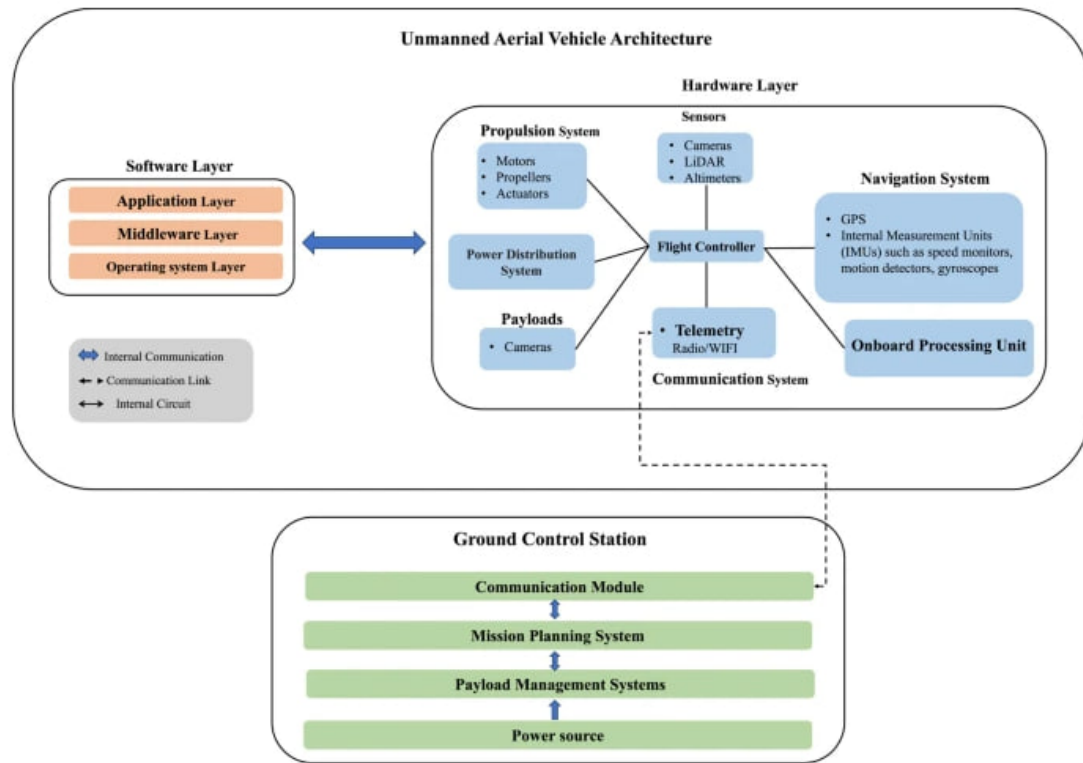


Figure 1.1: Unmanned Aerial Vehicle architecture.

1.2.2 Unmanned Aerial System (UAS)

A UAV alone does not constitute a complete operational system. The broader concept of an Unmanned Aerial System (UAS) encompasses all components required for autonomous aerial operations. A typical UAS consists of five main components [2]:

(1) UAVs: The most crucial component of the UAS. The capacity, performance, and operational theatre of the entire system are directly dependent on the UAV's capabilities, such as payload, altitude, range, and endurance.[2]

(2) Payloads: Equipment added to the UAV for carrying out different operational missions, including sensors, cameras, and other specialised devices.[2]

(3) Command and Control Systems: Static or portable ground control stations (GCS), air traffic control stations, or ground data terminals that facilitate interaction between human operators and UAVs.[2]

(4) Support Equipment: Launchers and other ground equipment that provide the UAV with its flight speed over an extremely short distance and time.[2]

(5) Control and Data Transmission Links: Communication links that define how information is sent and received between UAVs and the GCS, enabling real-time command and telemetry exchange.[2]



Figure 1.2: Components of a typical Unmanned Aerial System

1.2.3 Types and Characteristics of UAVs

UAVs exhibit a wide diversity in physical design, operational range, and performance characteristics. Understanding these dimensions is essential for selecting appropriate communication and security strategies. The following subsections present the principal classification axes [3].

(a) Range and Altitude : The range of a UAV is defined by the maximum distance at which it can be remotely controlled, typically from a few hundred metres to several kilometres for small UAVs. Altitude refers to the highest height a UAV can reach, often limited by regulations. UAVs adjust their position to improve coverage and meet Quality of Service (QoS) requirements. The maximum altitude is used to classify aerial systems into two main groups[3] :

Low-Altitude Platforms (LAPs) : are relatively low-cost and can be deployed quickly. They are mainly used for short-range line-of-sight (LoS) communication, improving network performance and local connectivity.

High-Altitude Platforms (HAPs): operate at much higher altitudes and can cover larger areas while staying airborne for longer periods. However, they are more complex to implement and are typically used to provide internet access to wide or underserved regions.

(b) Flying Mechanism : UAVs are classified into three types according to their flying mechanics [3]:

Rotary-Wing UAVs (multirotors): are capable of Vertical Take-Off and Landing (VTOL) and can hover in place. Their exceptional agility makes them well-suited for supporting wireless communication and serving as aerial base stations. However, they exhibit higher power consumption due to constantly counteracting gravity.

Fixed-Wing UAVs : are distinguished by their ability to glide through the air, offering enhanced power efficiency and the capacity to transport larger payloads at higher speeds. They require a runway for take-off and landing and cannot hover over a specific location. Despite their efficiency advantages, they are generally more expensive than rotary-wing UAVs.

Hybrid-Wing : UAVs combine features of both fixed-wing and rotary-wing designs.

(c) Speed and Flight Time

Flight time, or longevity, refers to the duration a UAV can remain airborne without re-

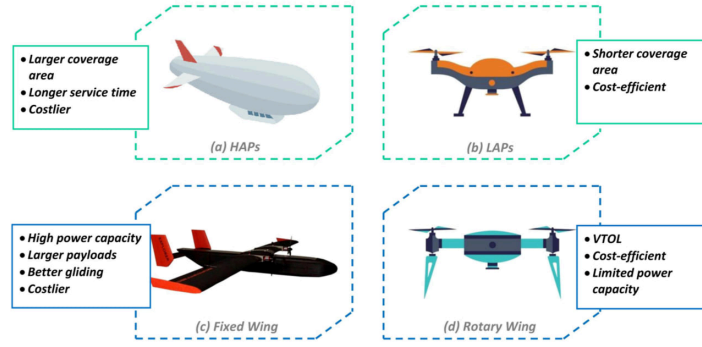


Figure 1.3: Types of Unmanned Aerial System

fuelling or recharging. Small commercial UAVs typically fly for 20–30 minutes, whereas larger platforms can fly for several hours. Technological advancements have significantly extended endurance; the Skyfront Tailwind UAV achieves nearly two hours of hovering using a hybrid-electric power source, while the Skyfront Perimeter-8 has demonstrated approximately 13 hours of flight with a gasoline-electric hybrid system. Despite these advances, limited endurance remains a practical constraint for widespread deployment in wireless networks [3].

(d) Payload :

The payload of a UAV, which assesses its lifting capacity, refers to the heaviest load it can carry. UAV payloads range from hundreds of grams to several hundred kilograms. Payload capacity determines the quantity and quality of sensors, communication equipment, and accessories a UAV can transport. A higher payload comes at the cost of larger dimensions, higher power requirements, and shortened flight duration. Common payloads include visual recording equipment and sensors. When used as aerial base stations or remote radio heads (RRHs), UAVs may require payloads weighing several kilograms [3].

(e) Power Supply : A UAV's power source directly affects its flight time. Most commercial UAVs use rechargeable batteries, while larger ones may rely on solar energy or combustion fuel for longer endurance. For UAV-based base stations, the power supply must support both propulsion and onboard equipment such as antennas and amplifiers, with at least 5 W required for transmission. The choice of power source depends on mission requirements and the trade-off between endurance and energy consumption [3].

1.2.4 Design Considerations of UAVs

Because of the unique features of UAVs, different planning considerations are required compared to standard network systems. The following seven considerations are central to any UAV network design [4]:

Adaptability: UAV nodes are highly mobile and their behaviour continuously evolves. Network paths and node separation vary over time, and UAV failures dynamically reduce

fleet size. System performance therefore depends critically on the ability to adapt to topology changes and node failures in real time [4].

Latency: Delay is one of the most fundamental design constraints. Time-critical applications such as pursuit and recovery demand minimal latency for real-time data exchange. Non-time-sensitive applications such as city mapping can tolerate higher latency values [4].

Mobility: UAV mobility plays an important role in system performance. Whether nodes follow a coordinated group pattern or operate independently, it is critical to ensure that mobility patterns are feasible for completing tasks successfully and on schedule [4].

UAV Platform Constraints: Due to limited payload capacity, UAV systems face hardware constraints on weight, size, and energy. A lighter payload extends flight duration but reduces available auxiliary resources. Space limitations also affect the integration of sensors and communication modules [4].

Scalability: Multi-UAV systems improve overall network performance compared to single-UAV deployments. Increasing fleet size enhances mission speed and system reliability, particularly for time-sensitive applications. Scalable designs are therefore essential [4].

Bandwidth Requirement: UAV applications such as surveillance and reconnaissance generate high-resolution video, images, and audio that must be relayed to the GCS under strict delay bounds. High transmission capacity is required to complete activities productively and on time [4].

Interoperability: UAV systems must be designed to operate seamlessly with heterogeneous networks and devices, ensuring integration with existing communication infrastructures [4].

1.3 Flying Ad-Hoc Networks (FANETs)

FANET is an Ad-Hoc network structure formed by a set of UAVs, of which at least one must be connected to a GCS or satellite. FANETs differ from existing Ad-Hoc networks such as MANETs (Mobile Ad-Hoc Networks) and VANETs (Vehicular Ad-Hoc Networks), though they can be viewed as a specialised form of these. The key distinguishing factor lies in the unique operational characteristics imposed by the aerial nature of UAVs [2]

1.3.1 FANET Characteristics

Flying Ad-Hoc Networks exhibit distinct operational characteristics that fundamentally influence their design and security requirements. These characteristics are detailed below [2]:

Network Topology: In FANETs, the network topology changes frequently due to the high degree of mobility of UAVs. This dynamic topology makes routing and coordination significantly more challenging than in terrestrial networks.

Mobility Models: In some mobility models, the flight plan is predetermined and the map is recalculated at each change. Other models employ random mobility — using random speed and directions — reflecting unstructured autonomous missions.

Node Mobility: The degree of mobility of UAVs is considerably higher than that of MANET and VANET nodes. UAVs may change their position in a much shorter time interval, which can cause communication disruptions between nodes.

Node Density: In FANET networks, nodes must be sparsely distributed in the sky with large distances between them, according to the nature of flight. Node density in FANETs is therefore much lower than in MANET and VANET.

Radio Propagation Model: Unlike MANET, FANETs rely on Line-of-Sight (LoS) communication between UAVs and between UAVs and the GCS, which is both an advantage for link quality and a constraint on network topology.

Localization: It is challenging to determine the accurate position of UAVs in a FANET due to their high mobility. Localization data must be updated in shorter time intervals to maintain accurate situational awareness.

Power Consumption: In FANETs, communication hardware is powered by the UAV's own energy source. Large UAVs are not power-sensitive in this respect, contrarily to MANET nodes. However, mini UAVs have very limited electrical capacity, making power management a critical concern.

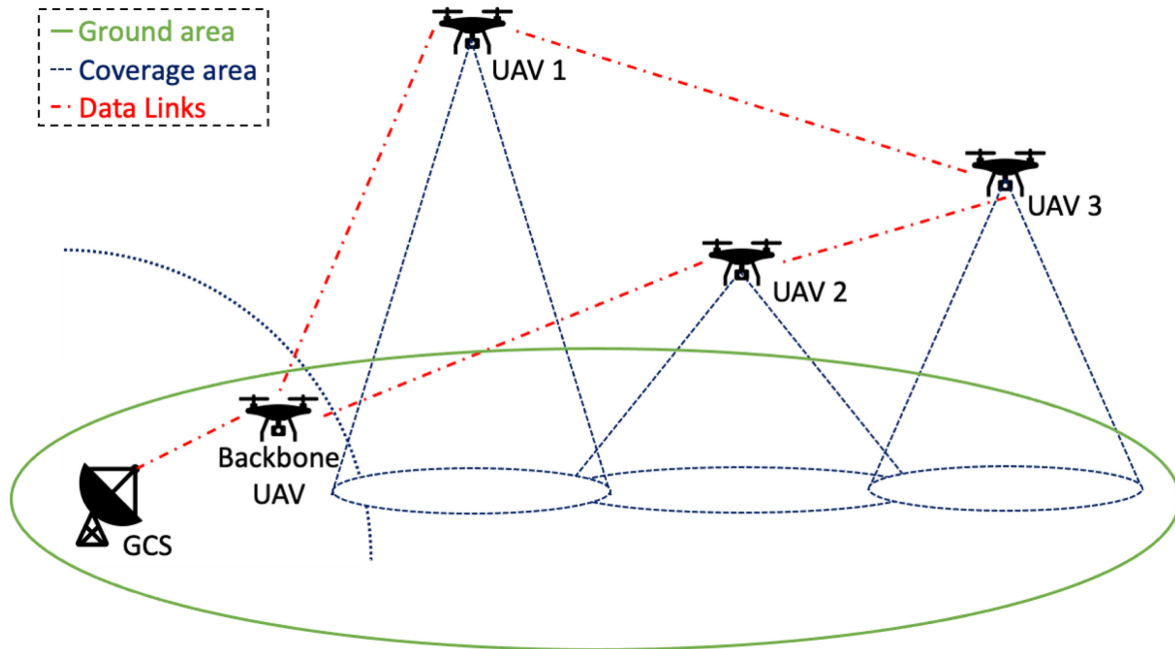


Figure 1.4: Extend coverage area by using FANET.

1.4 UAV Communication

Effective communication is the backbone of UAV operations, enabling real-time control, telemetry exchange, and cooperative mission execution. UAV communication systems are characterised by both the network architecture, the structural topology through which nodes exchange data, and the protocols that govern the format and rules of that exchange. Understanding these two dimensions is crucial for designing robust and secure UAV systems [5].

1.4.1 UAV Communication Architectures

UAV communication architectures are broadly divided into centralised and decentralised models, each offering distinct trade-offs in terms of reliability, scalability, and resilience to failure [5].

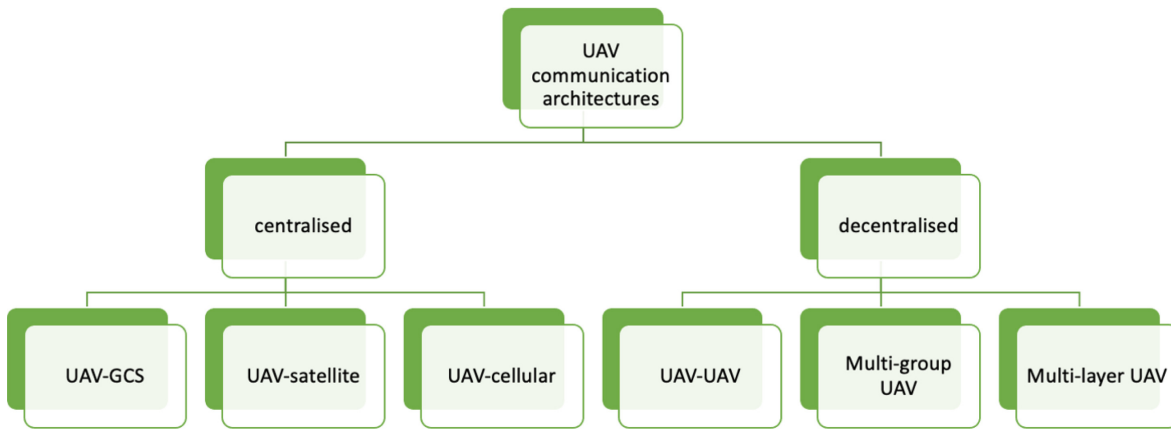


Figure 1.5: UAV Communication Architecture Taxonomy.

(A) Centralised Architectures

In a centralised architecture, all UAVs communicate through a single central controller, inherently introducing a single point of failure. Three variants exist. In the UAV-GCS model, every UAV must directly connect to the GCS to obtain data; this is not advisable in changeable environments such as stormy weather. In the UAV-Satellite model, communication is relayed through a satellite, which is suitable when the distance between the GCS and UAVs is large. In the UAV-Cellular model, communication is performed via cellular technology, using base stations to implement routing that facilitates communication between nodes [5].

(B) Decentralised Architectures

In a decentralised architecture, UAVs communicate directly or indirectly with the GCS without relying on a single central node. Backbone UAVs act as gateways, relaying data between the GCS and subordinate UAVs. FANETs represent the most flexible form, where

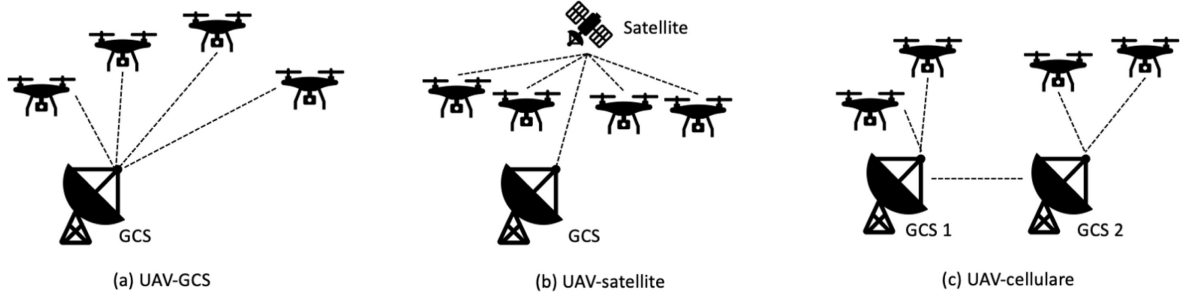


Figure 1.6: Centralised UAV Communication Architectures.

UAVs cooperate without central coordination and the network structure is not disrupted by nodes joining or leaving [5].

Three decentralised topology variants are identified. In the **UAV Ad-Hoc Network**, a backbone UAV uses high power for long-range GCS access while communicating to subordinates via short-range links, extending coverage and suited for small UAVs with lightweight transceivers. **The Multi-Group UAV Ad-Hoc Network** partitions UAVs into independent groups, each operating as a FANET with a backbone UAV mediating inter-group communication; intra-group exchanges bypass the GCS, though the GCS remains a potential single point of failure. **The Multi-Layer UAV Ad-Hoc Network** introduces a tiered topology where a lower layer handles intra-group communication and an upper layer enables backbone UAVs to communicate with peers or the GCS; inter-group exchange does not rely on the GCS, mitigating throughput bottlenecks and enabling high scalability and fault tolerance [5].

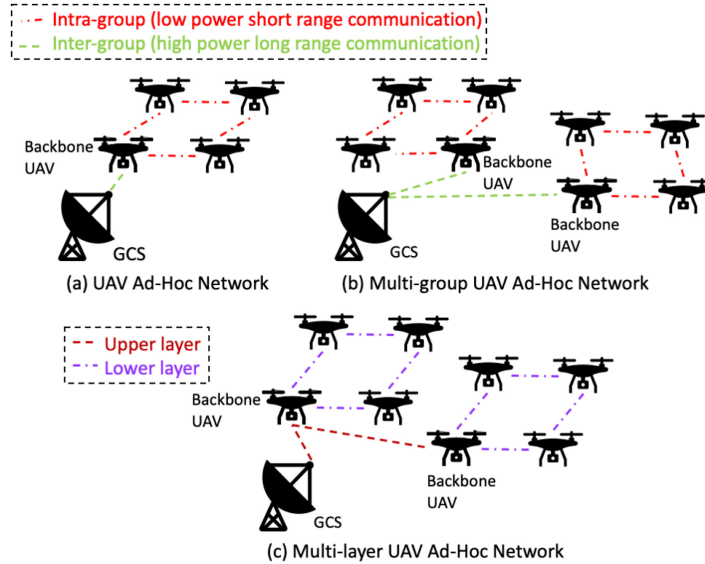


Figure 1.7: Decentralised UAV communication architectures.

1.5 Taxonomy of UAV Attack Types

The rapid growth of UAV technology has introduced new security challenges and increased the number of potential cyber threats targeting UAV systems. Attackers may exploit vulnerabilities to affect the confidentiality, integrity, and availability of UAV operations. Therefore, understanding the UAV threat landscape is essential for developing efficient detection and protection mechanisms [6]. The following taxonomy classifies UAV-related attacks into five main categories according to the system layer or resource being targeted [6].

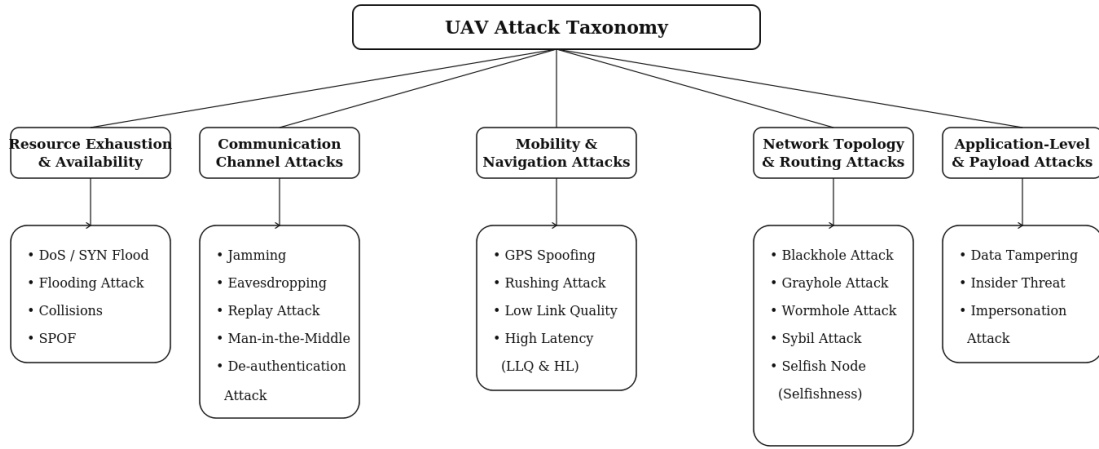


Figure 1.8: UAV Attack Taxonomy.

1.5.1 Resource Exhaustion and Availability Attacks

Resource exhaustion and availability attacks target the limited computational power and communication resources inherent to UAV systems. These attacks disrupt UAV operations by overwhelming computational or communication capabilities [6].

- **Denial-of-Service (DoS):** DoS attacks flood UAV nodes or communication channels with excessive packets, depleting processing power and bandwidth until the node becomes unresponsive. A notable variant is the SYN Flood, which abuses the TCP handshake by sending large numbers of SYN packets, exhausting connection tables and causing crashes [6].
- **Flooding Attack:** Flooding attacks overwhelm UAVs with excessive data packets, consuming memory buffers and processing capacity, degrading communication reliability, and delaying legitimate data transfers critical for mission success [6].

- **Collisions:** Collisions occur when multiple UAV nodes transmit simultaneously on the same frequency, causing packet loss and retransmissions. The high mobility and dynamic topology of FANETs significantly increase collision likelihood [6].
- **Single Point of Failure (SPOF):** Single Point Of Failure arises when a critical UAV or GCS node fails or is compromised, causing operational breakdown. This vulnerability is particularly acute in centralised UAV architectures [6].

1.5.2 Communication Channel Attacks

Communication channel attacks aim to intercept, manipulate, or disrupt data transmission between UAV nodes or ground control systems, compromising the confidentiality, integrity, or availability of transmitted data [6].

- **Jamming (Active Interference):** Intentional transmission of interference signals disrupts legitimate UAV communication channels, causing significant degradation of control signals and real-time data exchange [6].
- **Eavesdropping:** Eavesdropping intercepts UAV communications to gather sensitive or classified data, violating confidentiality and enabling further attacks such as replay or impersonation.
- **Replay Attack:** Captured legitimate packets are retransmitted fraudulently at a later time to mislead UAV or control stations, enabling unauthorised access or system manipulation .
- **Man-in-the-Middle (MITM):** MITM attacks intercept and potentially alter UAV communications in real time, compromising message confidentiality, integrity, and authenticity .
- **De-authentication Attack:** Attackers send falsified de-authentication packets to forcibly disconnect legitimate UAV nodes or control stations, creating operational instability .

1.5.3 Mobility and Navigation Attacks

These attacks specifically target UAV positioning and movement controls, affecting flight trajectory, path planning, and collision avoidance [6].

- **GPS Spoofing:** Attackers transmit counterfeit GPS signals to deceive UAV navigation systems, causing incorrect location reporting or complete navigational failure, critically compromising UAV safety [6].

- **Rushing Attack:** Route discovery packets are rapidly forwarded to establish unauthorised control over network routing, disrupting legitimate navigation paths and degrading routing protocol reliability .
- **Low Link Quality and High Latency (LLQ and HL):** Caused by intentional interference or network congestion, these conditions produce high communication latency and poor signal quality, critically impacting real-time navigation and coordination among UAVs .

1.5.4 Network Topology and Routing Attacks

These attacks compromise the integrity and reliability of routing protocols and UAV network topology, often exploiting vulnerabilities within decentralised architectures.

- **Blackhole Attack:** Malicious nodes falsely advertise optimal routes to attract network traffic, then drop all received packets, disrupting UAV network reliability and operational performance [6].
- **Grayhole Attack:** Similar to the blackhole attack, grayhole attackers selectively and unpredictably forward or drop packets, complicating detection and causing significant routing instability [6].
- **Wormhole Attack:** Attackers create a covert communication tunnel between malicious nodes, secretly intercepting and redirecting packets, potentially leading to severe data breaches [6].
- **Sybil Attack:** An attacker creates multiple fake UAV identities within the network, undermining trust mechanisms, routing decisions, and collaborative mission integrity [6].
- **Selfish Node (Selfishness):** Selfish nodes intentionally limit their packet forwarding to conserve resources, disrupting network cooperation and decreasing overall UAV network performance [6].

1.5.5 Application-Level and Payload Attacks

These attacks compromise UAV payload or application-layer services, affecting data integrity, confidentiality, and operational trustworthiness [6].

- **Data Tampering and Modification:** Attackers manipulate transmitted payload data or application messages to mislead UAV operations, degrading mission accuracy and reliability .

- **Insider Threat:** Authorised insiders intentionally misuse their legitimate access to leak sensitive information or manipulate UAV operations, posing severe internal security threats .
- **Impersonation Attack:** Attackers disguise themselves as legitimate UAV nodes or ground stations to gain unauthorised access, manipulate data exchanges, or mislead operational decisions .

1.6 Intrusion Detection Systems (IDS)

Given the diverse and sophisticated threat landscape described in the previous section, intrusion detection constitutes a critical layer of defence for UAV networks. This section provides a formal definition, a structured three-axis classification, a review of AI-based detection approaches specifically applied to UAV networks, and an analysis of IDS deployment strategies[7].

1.6.1 Definition and Goal of an IDS

An intrusion can be defined as any kind of unauthorised activity that causes damage to an information system. An Intrusion Detection System (IDS) is a software or hardware system that identifies malicious actions on computer systems in order to allow system security to be maintained. The fundamental goal of an IDS is to identify different kinds of malicious network traffic and computer usage that cannot be identified by a traditional firewall, enabling timely response and mitigation [7] .

1.6.2 IDS Classification

Intrusion Detection Systems can be classified along three orthogonal axes: the technology used for monitoring, the methodology employed for detection, and the analytical approach used to reason about collected data. The following subsections examine each axis in detail [8].

IDS Technologies

IDS technologies are classified according to the source and scope of the data they monitor. Four main technology categories are identified :

- **Network-Based IDS (NIDS):** A NIDS monitors network traffic for the security of network devices, analysing protocols at the network, application, and transport layers to detect suspicious activities.

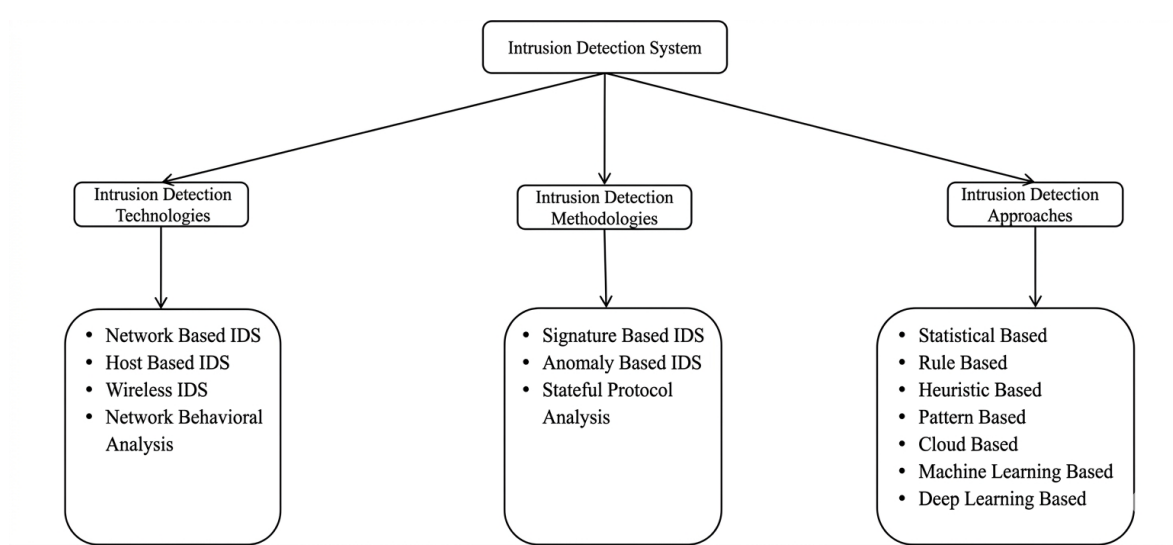


Figure 1.9: Intrusion Detection Systems Classification

- **Host-Based IDS (HIDS):** A HIDS monitors a host's properties and activities — including traffic information, system logs, and file access — to detect potential threats originating from or affecting a specific node.
- **Wireless IDS (WIDS):** A WIDS monitors wireless network traffic and analyses wireless protocols to identify suspicious activity. It is widely deployed across the coverage area of a wireless network but cannot inspect upper-layer protocols such as TCP or UDP.
- **Network Behavioural Analysis:** A Network Behavioural Analysis system examines network traffic statistics to identify unusual flows such as DDoS attacks, malware (worms, backdoors), and policy violations. NBA sensors either sniff packets directly or consume network flow information from routers and network devices.

IDS Methodologies

IDS methodologies define the detection strategy — the principle by which the system distinguishes malicious from benign activity. Three main methodologies are recognised :

- **Signature-Based IDS:** A signature is a pattern corresponding to a known attack. Signature-based detection compares observed events against a database of known attack signatures. It is highly effective for known threats but largely ineffective against previously unseen attacks or novel variants.
- **Anomaly-Based IDS:** Anomaly-based detection compares observed activities against a model of normal behaviour. Deviations from the established baseline are flagged as potential intrusions. This approach can detect unknown attacks but may produce higher false positive rates.

- **Stateful Protocol Analysis:** Stateful protocol analysis compares observed protocol activity against predefined profiles of normal protocol behaviour for each state. Unlike anomaly-based detection, it relies on universal protocol profiles rather than host-specific baselines, and tracks the state of network, transport, and application protocols over time.

IDS Approaches

IDS approaches describe the algorithmic and analytical techniques used to implement the chosen methodology. Seven principal approaches are identified in the literature :

- **Statistical-Based:** Monitors normal transactions to build a legitimate profile. A score is assigned per transaction and compared against a threshold; exceeding it triggers an alarm. Statistical metrics such as mean, median, variance, and standard deviation are used to model normal behaviour.
- **Rule-Based:** Applies a set of predefined rules for detecting intrusions. Artificial intelligence may be used to derive rules automatically from known attack patterns. A small set of rules can recognise many attack variants, making this approach more compact than signature-based methods.
- **Heuristic-Based:** Searches for intrusions based on known malicious behavioural patterns. The system builds a model of acceptable behaviour and flags deviations.
- **Pattern-Based:** Extracts meaningful patterns from collected data and identifies attacks by matching against known malicious patterns. Detects known attacks efficiently but struggles with zero-day threats whose signatures are not yet known.
- **Cloud-Based:** Leverages cloud computing infrastructure to identify malicious network activities from multiple perspectives, overcoming classical single-node detection limitations through aggregated analysis.
- **Machine Learning-Based:** Automates the analysis process using supervised, unsupervised, or semi-supervised learning. Primary advantages include adaptability, high performance, flexibility, and the ability to detect novel attack types.
- **Deep Learning-Based:** A subfield of machine learning that learns hierarchical representations from examples, often eliminating manual feature engineering. Can detect anomalies in large-scale, multi-dimensional, and dynamically changing data.

1.6.3 IDS Deployment Strategies

The deployment strategy of an IDS in UAV networks significantly influences its operational effectiveness, response time, scalability, and robustness against cyber threats. Deployment refers to the organisational and functional architecture that handles how intrusion detection tasks are distributed or centralised among UAV nodes and associated GCS [6]. Proper de-

ployment strategy selection is critical given the unique constraints of UAV systems, limited computational resources, real-time responsiveness requirements, and the need for robustness against diverse threat types [6]

Centralised Deployment

In centralised IDS deployment, intrusion detection operations — including analysis and decision-making — are primarily conducted at a single central location, usually a GCS or dedicated central node. UAVs act mainly as data collection points, transmitting gathered data to the centralised node for comprehensive analysis. This approach is also more feasible for heavier deep learning models. It simplifies data management and ensures powerful computing resources can be employed, significantly enhancing analytical capability and accuracy [6].

Decentralised (Distributed) Deployment

In decentralised or distributed IDS deployments, detection capabilities are dispersed across individual UAV nodes rather than concentrated at a single central point. Each UAV operates autonomously or semi-autonomously, performing local data analysis and anomaly detection. This architecture significantly reduces latency by enabling immediate detection and response to cyber threats directly at the UAV node level [6].

Federated Deployment

Federated deployment combines aspects of centralised and decentralised architectures by enabling UAV nodes to collaboratively train intrusion detection models without sharing raw data. Each UAV node trains a local model using its own data, and periodic model updates are securely transmitted to a centralised aggregator — typically located at a GCS — which consolidates these updates into a global model and redistributes it back to individual UAVs [6]

1.7 Kolmogorov–Arnold Networks (KAN)

Kolmogorov–Arnold Networks (KAN) are a recently introduced class of neural architectures inspired by the Kolmogorov–Arnold representation theorem. Unlike classical neural networks that approximate functions using fixed nonlinear activations on nodes, KANs shift the learning capacity from nodes to edges by parameterizing learnable univariate functions [9].

1.7.1 Kolmogorov–Arnold Representation Theorem

The Kolmogorov–Arnold theorem states that any continuous multivariate function can be represented as a finite composition of continuous univariate functions [9].

Formally, for a continuous function

$$f : \mathbb{R}^n \rightarrow \mathbb{R},$$

there exist continuous functions $\phi_q(\cdot)$ and $\psi_{p,q}(\cdot)$ such that:

$$f(x_1, x_2, \dots, x_n) = \sum_{q=0}^{2n} \phi_q \left(\sum_{p=1}^n \psi_{p,q}(x_p) \right)$$

This theorem guarantees that high-dimensional functions can be decomposed into compositions of 1D functions, which forms the theoretical foundation of KANs [9].

1.7.2 From Theorem to KAN Architecture

Kolmogorov–Arnold Networks operationalize this theorem by replacing standard linear weights with learnable univariate functions [9].

A KAN layer is defined as:

$$y_j = \sum_{i=1}^n \phi_{i,j}(x_i)$$

where:

- x_i are input features,
- $\phi_{i,j}(\cdot)$ are learnable 1D functions,
- y_j is the output feature.

Unlike Multi-Layer Perceptrons (MLPs), where non-linearity is applied at nodes, KANs embed non-linearity directly on edges.

Each function $\phi_{i,j}$ is typically parameterized using spline-based representations:

$$\phi(x) = \sum_k c_k B_k(x)$$

where $B_k(x)$ are basis spline functions and c_k are learnable coefficients.

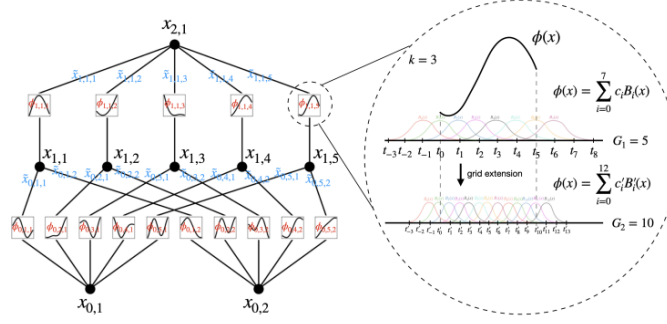


Figure 1.10: Notations of activations that flow through the network.

1.7.3 KAN vs MLP

The key differences between KAN and traditional MLP architectures are summarized as follows[9]:

Model	Multi-Layer Perceptron (MLP)	Kolmogorov-Arnold Network (KAN)
Theorem	Universal Approximation Theorem	Kolmogorov-Arnold Representation Theorem
Formula (Shallow)	$f(\mathbf{x}) \approx \sum_{i=1}^{N(c)} a_i \sigma(\mathbf{w}_i \cdot \mathbf{x} + b_i)$	$f(\mathbf{x}) \approx \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^n \phi_{q,p}(x_p) \right)$
Model (Shallow)	(a) fixed activation functions on nodes learnable weights on edges	(b) learnable activation functions on edges sum operation on nodes
Formula (Deep)	$\text{MLP}(\mathbf{x}) = (\mathbf{W}_3 \circ \sigma_2 \circ \mathbf{W}_2 \circ \sigma_1 \circ \mathbf{W}_1)(\mathbf{x})$	$\text{KAN}(\mathbf{x}) = (\Phi_3 \circ \Phi_2 \circ \Phi_1)(\mathbf{x})$
Model (Deep)	(c) $\mathbf{W}_3$ σ_2 $\mathbf{W}_2$ σ_1 $\mathbf{W}_1$ $\mathbf{x}$ nonlinear, fixed linear, learnable	(d) Φ_3 Φ_2 Φ_1 $\mathbf{x}$ nonlinear, learnable linear, learnable

Figure 1.11: Multi-Layer Perceptrons (MLPs) vs. Kolmogorov-Arnold Networks (KANs).

- **MLP:** Learns weights on edges and applies fixed activation functions on nodes.
- **KAN:** Learns activation functions on edges, replacing fixed nonlinearities.

Mathematically, an MLP layer is:

$$y = \sigma(Wx + b)$$

while a KAN layer is:

$$y_j = \sum_i \phi_{i,j}(x_i)$$

This structural difference allows KANs to achieve higher interpretability and improved ap-

proximation efficiency for certain function classes [9].

1.7.4 Why KAN for UAV Intrusion Detection

In UAV intrusion detection systems, the feature space is highly nonlinear, heterogeneous, and sensitive to small perturbations in communication behavior. Traditional MLP-based classifiers may require deeper architectures to approximate such functions [9].

KANs are particularly suitable for this context due to:

- Their ability to model complex nonlinear relationships using explicit function decomposition.
- Improved interpretability of learned feature transformations.
- Better approximation efficiency in low-data or imbalanced scenarios.

As demonstrated in the original KAN formulation, replacing fixed activations with learnable functions improves representation flexibility while maintaining compact model structures, which is crucial for UAV systems with computational constraints [9].

1.8 Conclusion

This chapter provided a comprehensive overview of the fundamental concepts related to UAV networks and their security requirements. It covered UAV architectures, FANET properties, and communication models, emphasizing the challenges introduced by mobility, scalability, and resource constraints.

It also presented a detailed taxonomy of UAV cyberattacks, illustrating the diversity of threats across network, communication, navigation, routing, and application layers. In addition, Intrusion Detection Systems were analyzed in terms of their technologies, methodologies, and deployment strategies in centralized and distributed environments.

Finally, Kolmogorov–Arnold Networks were introduced as an emerging learning approach capable of modeling complex nonlinear behaviors, making them suitable for UAV intrusion detection tasks.

Overall, the chapter establishes the theoretical foundation required for the proposed architecture and implementation in the following chapters.

Chapter 2

Chapter 2: Related Work

2.1 Introduction

This chapter reviews existing research on intrusion detection systems (IDS) for UAV networks. It covers different approaches proposed in the literature, including network-based and system-based IDS, as well as their deployment architectures such as centralized and distributed models. The chapter also discusses optimization strategies for resource-constrained UAV environments and highlights various detection methodologies, including behavioral and rule-based techniques. Finally, it presents a comparative discussion of the reviewed works to outline their main contributions and limitations.

2.2 Different intrusion detection systems for UAVs security

Securing Unmanned Aerial Vehicles (UAVs), Flying Ad-Hoc Networks (FANETs), and the Internet of Drones (IoD) has emerged as a critical research domain due to the increasing reliance on autonomous aerial platforms in civilian, industrial, and military applications. However, the constrained computational resources, limited onboard energy, dynamic mobility, and distributed nature of UAV environments introduce significant challenges for intrusion detection systems (IDS). Consequently, existing UAV IDS research seeks to achieve a balance between detection accuracy, computational efficiency, scalability, and real-time responsiveness.

To provide a structured understanding of the literature, existing IDS approaches for UAV networks can be analysed according to multiple complementary dimensions, including the source of monitored data, deployment architecture, optimization strategy, and detection methodology.

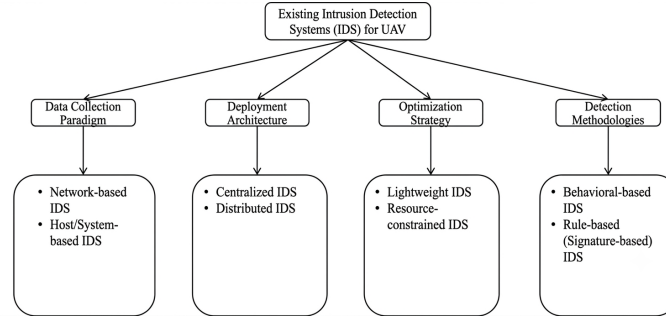


Figure 2.1: Intrusion Detection Systems (IDS) Taxonomy in UAV Networks

2.2.1 Data Collection Paradigms

(A) Network-Based IDS

Network-Based Intrusion Detection Systems (NIDS) represent the most widely adopted category in UAV security research. These approaches monitor communication traffic, telemetry streams, routing behavior, and packet exchanges between UAV nodes and ground control stations in order to identify malicious activities.

Al-Fuwaires and Mishra [10] developed a machine learning-based detection pipeline using conventional classifiers such as Random Forest and Decision Trees trained on the NSL-KDD dataset. Although the proposed approach achieved an accuracy exceeding 97%, its reliance on traditional network datasets limits its ability to accurately represent realistic UAV communication environments.

To address the scarcity of authentic drone datasets, Whelan et al. [11] proposed the MAVIDS framework, which applies unsupervised one-class classification and Principal Component Analysis (PCA) directly to PX4 flight logs. Their framework successfully detected anomalies including GPS spoofing and communication jamming without requiring labelled datasets.

Several studies have focused on handling class imbalance problems commonly encountered in intrusion detection datasets. Bouhamed et al. [12] introduced a Deep Reinforcement Learning (DRL) framework that employs a reward optimization mechanism to prioritize minority attack classes within the CICIDS2017 dataset. Similarly, Gamal et al. [13] combined Long Short-Term Memory (LSTM) networks with the Synthetic Minority Over-sampling Technique (SMOTE) to improve temporal attack classification. Hamad et al. [14] further enhanced sequential attack modelling by proposing an Extended LSTM (XLSTM) architecture integrated with Recursive Feature Elimination (RFE) to reduce noisy and redundant features.

Other works have explored architectural improvements for network-level intrusion detection. The researchers in [15] developed UAV-IDS-ConvNet, a convolutional deep learning architecture evaluated on the UAV-IDS-2020 dataset containing encrypted drone Wi-Fi traffic. Wang et al. [16] improved multiclass threat detection through the integration of

group convolution networks and snapshot ensemble learning. Furthermore, Mehmood et al. [17] demonstrated the practical feasibility of deploying lightweight machine learning models within wireless sensor network environments using the COOJA simulation platform.

(B) System-Based IDS

Unlike network-based approaches, System-Based Intrusion Detection Systems monitor the internal operational state of UAV platforms, including onboard sensors, flight controllers, electrical components, and hardware-level signals. These approaches are particularly effective for detecting attacks that cannot be directly observed through packet inspection alone.

Rahman et al. [18] proposed a three-layered UAV security architecture targeting flight controllers and power electronic subsystems. Their framework combines Support Vector Machines (SVM) with rule-based signal analysis to detect hardware Trojan attacks and False Data Injection (FDI) attacks affecting onboard sensor measurements.

Alheeti et al. [19] utilized ICMetric technology to extract unique physical fingerprints from micro-electromechanical system (MEMS) sensors. These hardware-level signatures were then combined with network traffic information using a Deep Neural Network (DNN) to improve intrusion detection accuracy. To address the severe resource limitations of lightweight UAV platforms, Khalil and Rahman [20] implemented an Adaptive Neuro-Fuzzy Inference System (ANFIS) directly on an Arduino Nano microcontroller. Their lightweight architecture enabled real-time classification of flight telemetry while requiring minimal memory and computational resources.

2.2.2 Deployment Architectures

(A) Distributed IDS

Distributed IDS architectures aim to eliminate single points of failure by enabling cooperative intrusion detection across multiple UAV nodes. These systems typically rely on decentralized communication, collaborative decision-making, and distributed intelligence mechanisms.

Abu Al-Haija and Al Badawi [15] proposed a peer-to-peer collaborative detection framework where UAV nodes independently monitor local conditions while maintaining trust synchronization using blockchain-based consensus mechanisms. This approach improves resilience against centralized failures while enhancing network trust management.

To preserve operational privacy in swarm environments, da Silva et al. [21] implemented a Federated Learning-based intrusion detection framework for UAV networks. Under this architecture, each UAV trains a local autoencoder model using private flight data and shares only updated model parameters with a central aggregator through the FedYogi optimization algorithm, thereby preventing raw data exposure.

(B) Centralized IDS

Centralized IDS architectures collect telemetry data and network traffic from distributed UAV nodes and process them using centralized edge servers, cloud systems, or ground control stations. These architectures provide higher computational capacity and simplified management at the cost of increased communication dependency and potential single points of failure.

Shrestha et al. [22] developed an integrated 5G-satellite gateway architecture that offloads computationally intensive detection tasks from UAV nodes to terrestrial and satellite-based edge servers. Their centralized Decision Tree-based framework demonstrated effective detection of multiple attack vectors while preserving onboard energy resources.

Similarly, Gamal et al. [13] employed a centralized architecture where data preprocessing, normalization, and deep LSTM training operations are executed at a base station rather than onboard UAVs. This design significantly reduces storage and computational burdens on aerial platforms.

2.2.3 Optimization Strategies

Lightweight IDS

Because UAVs operate under strict energy, storage, and processing constraints, lightweight IDS architectures constitute a major research direction within UAV cybersecurity.

Khalil and Rahman [20] proposed an ANFIS-based IDS requiring only 5 KB of memory and approximately 3 milliseconds of execution time on microcontroller-based hardware. Bouhamed et al. [12] further reduced onboard computational overhead through a periodic offline retraining mechanism where intensive learning operations are transferred to a ground station.

Whelan et al. [11] demonstrated that localized Autoencoder models could execute efficiently on NVIDIA Jetson Nano companion computers while processing thousands of predictions per second. Medhi et al. [23] advanced lightweight IDS optimization through UAV-DiPNID, which combines model pruning and knowledge distillation techniques to reduce model size by approximately 90% and decrease inference latency by more than 80% without significantly affecting detection accuracy.

2.2.4 Detection Methodologies

Behavioral and Rule-Based IDS

Behavioral and rule-based intrusion detection approaches focus on improving interpretability and decision transparency by utilizing structured logic, predefined behavioral profiles, and explicit reasoning mechanisms.

Xie et al. [24] introduced the Extended Belief Rule Base (EBRB) framework, which employs optimized IF-THEN rules derived from packet size distributions and traffic arrival intervals to detect anomalous behavior with limited training data requirements.

Ouiazane et al. [25] proposed a multi-agent architecture where a lightweight signature-based filtering agent first evaluates incoming traffic before activating more computationally intensive machine learning mechanisms when unknown patterns are encountered.

Additionally, Ihekoronye et al. [26] designed a Random Forest-based delay-tolerant IDS optimized through randomized search cross-validation for military UAV missions operating under severe payload and latency constraints.

2.2.5 Discussion

Despite significant advances in UAV intrusion detection research, several limitations remain insufficiently addressed. Many existing systems rely on generic network datasets that fail to accurately capture realistic UAV communication dynamics. Lightweight architectures often sacrifice detection performance to preserve energy efficiency, while centralized approaches remain vulnerable to communication bottlenecks and single points of failure. Furthermore, distributed IDS frameworks continue to face challenges related to coordination overhead, scalability, adaptive learning, and real-time decision synchronization.

These limitations highlight the need for efficient distributed intrusion detection architectures capable of combining high detection accuracy, lightweight operation, adaptive intelligence, and scalable cooperative detection mechanisms. Motivated by these challenges, the following section examines the E-DIDS architecture, which serves as the foundational framework adopted and extended in this thesis.

Reference	Category	Architecture	Technique	Main Contribution
Al-Fuwaiers and Mishra (2024)	Network-Based	Centralized	Random Forest, Decision Tree	Machine learning IDS for UAV IoT security using NSL-KDD dataset.
Whelan et al. (2022)	Network-Based	Lightweight	PCA, One-Class Classification	MAVIDS framework for anomaly detection using PX4 flight logs.
Bouhamed et al. (2021)	Network-Based	Lightweight	Deep Reinforcement Learning	Improved minority attack detection through reward-based learning.
Gamal et al. (2024)	Network-Based	Centralized	LSTM + SMOTE	Enhanced intrusion detection through temporal attack modelling.
Hamad et al. (2025)	Network-Based	Centralized	XLSTM + RFE	Deep learning IDS with feature selection and improved sequence learning.
Abu Al-Haija and Al Badawi (2024)	Network-Based	Distributed	Deep Learning	Collaborative real-time IDS for UAV networks.
Wang et al. (2021)	Network-Based	Centralized	Group CNN + Snapshot Ensemble	Improved multiclass intrusion classification performance.
Mehmood et al. (2022)	Network-Based	Lightweight	Machine Learning	Simulation-based IDS deployment using COOJA.
Rahman et al. (2021)	System-Based	Distributed	SVM + Rule-Based Analysis	Detection of hardware Trojans and false data injection attacks.
Alheeti et al. (2023)	System-Based	Centralized	ICMetric + DNN	Hybrid IDS combining hardware fingerprints and network traffic.
Khalil and Rahman (2023)	System-Based	Lightweight	ANFIS	Real-time lightweight IDS for resource-constrained UAVs.
da Silva et al. (2023)	Distributed IDS	Distributed	Federated Learning	Privacy-preserving collaborative anomaly detection for UAV swarms.
Shrestha et al. (2021)	Centralized IDS	Centralized	Decision Tree	Offloaded intrusion detection through 5G and satellite infrastructure.
Medhi et al. (2025)	Lightweight IDS	Lightweight	Pruning + Distillation	Reduced model size and inference latency while preserving accuracy.
Xie et al. (2022)	Behavioral IDS	Centralized	Belief Rule Base	Interpretable anomaly detection using expert-defined rules.
Ouiazzane et al. (2020)	Behavioral IDS	Distributed	Multi-Agent System	Hierarchical detection combining signatures and anomaly analysis.
Ihekoronye et al. (2022)	Behavioral IDS	Lightweight	Random Forest	Delay-tolerant IDS for military UAV environments.

Table 2.1: Comparative summary of representative intrusion detection systems for UAV security.

2.3 Reference Architecture : E-DIDS

2.3.1 Overview of E-DIDS

The Exhaustive Distributed Intrusion Detection System m for UAVs attacks detection and security enforcement(E-DIDS) (reference) framework represents a paradigm shift in UAV intrusion detection, transitioning from the conventional centralized detection model toward a fully distributed, cooperative, and subsystem-aware security architecture. The fundamental premise of E-DIDS is that no single IDS unit can adequately monitor the heterogeneous attack surface of a modern UAV system, which encompasses sensor layers, communication protocols, core system processes, and network interfaces simultaneously. By distributing detection responsibilities across specialized IDS units , each anchored to a distinct UAV subsystem . The framework achieves both detection granularity and systemic resilience that centralized approaches inherently cannot provide.[27]

2.3.2 E-DIS Proposed Scheme

E-DIDS Architecture

The E-DIDS architectural design decomposes the detection infrastructure into five functionally distinct yet tightly integrated modules. This modular decomposition is not merely an engineering convenience; it reflects a principled separation of concerns, where in each module encapsulates a specific aspect of the detection lifecycle , from raw data ingestion to threat-response actuation.[27]

(1) The Sensor Module : constitutes the perceptual layer of the framework. It is responsible for the continuous acquisition and preliminary assessment of operational data across spatially distributed points within the UAV system. Each sensor unit is configured to observe a designated subsystem component, ensuring that the detection coverage is both exhaustive and non-overlapping.[27]

(2) The Communication Module : governs the transmission of processed intelligence between the distributed IDS units and the Central Control Module. Its layered design comprising a transport layer for link establishment and a nucleus layer for content encapsulation , ensures that inter-IDS data exchange is both reliable and semantically structured. [27]

(3) The Central Control Module : serves as the analytical and decisional apex of the E-DIDS hierarchy. Upon receiving detection analytics from all subsystem IDSs, it performs a global threat synthesis, categorizes the severity of detected intrusions, and initiates the appropriate response protocol. Its internal composition , an analysis engine, a central console, and a response layer , reflects the sequential logic of detection, interpretation, and actuation. [27]

(4) The Management Module : addresses the operational governance of the distributed IDS network. It maintains configuration consistency across all subsystem IDSs, enforces security policy uniformity, and generates performance monitoring reports. Crucially, it functions as a distributed small engine, meaning each linked subsystem hosts its own management instance rather than deferring entirely to a central authority. [27]

(5) The Reporting Module : closes the feedback loop by aggregating detection outputs and producing structured incident reports. It distinguishes between the data aggregation function and the response initiation function through its two constituent layers, thereby ensuring that alarm generation and data persistence are decoupled.[27]

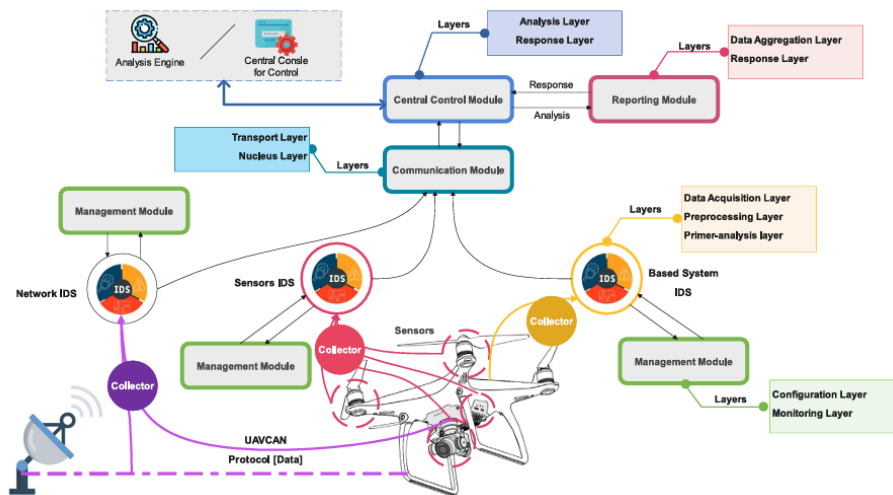


Figure 2.2: Diagram scheme of the E-DIDS modules.

E-DIDS behavioral study case

The E-DIDS framework defines UAV security through behavioral standards that describe the normal operational ranges of each UAV subsystem (sensors, communication, and flight control). Instead of treating the UAV as a single unit, the system models each subsystem separately because each one exhibits distinct behavioral patterns.

These behaviors are represented as a set of states where the UAV operates in either [27]:

- Normal behavior set N

- Abnormal (attack/fault) behavior set A

such that:

$$N \cup A = U \quad (2.1)$$

where U is the full universe of UAV operational states.

Behavior Representation

Each UAV state is mapped into a structured representation space:

$$R : U \rightarrow U_R \quad (2.2)$$

This mapping transforms raw flight events into feature vectors of behavioral indicators, forming:

- N_R : normal mapped states
- A_R : abnormal mapped states

These vectors encode subsystem conditions, including sensor readings, protocol states, and flight behavior constraints.

Detection Model

The intrusion detection module is defined as:

$$D = (f, l) \quad (2.3)$$

where:

$$f : \text{detection function} \quad (2.4)$$

$$l : \text{UAV flight log sequence} \quad (2.5)$$

The detection function operates as:

$$f : U \times R \rightarrow [0, 1] \quad (2.6)$$

It evaluates the similarity between observed behavior $s \in U_R$ and learned normal patterns S .

Decision Rule

A threshold-based classification is used:

$$D(s) = \begin{cases} \text{Normal,} & \text{if } f(s, S) \geq \varphi \\ \text{Attacked,} & \text{if } f(s, S) < \varphi \end{cases} \quad (2.7)$$

where:

$$\varphi : \text{detection threshold} \quad (2.8)$$

$$S : \text{stored safe behavioral patterns} \quad (2.9)$$

This means the system flags a UAV as compromised when its behavior deviates significantly from learned safe standards.

2.3.3 System model and proposed scheme

Implementing an exhaustive DIDS architecture using AI could involve several phases, including data collection and preprocessing, building and connection, training, and integrating the different IDS modules to run the central control of the E-DIDS.[27]

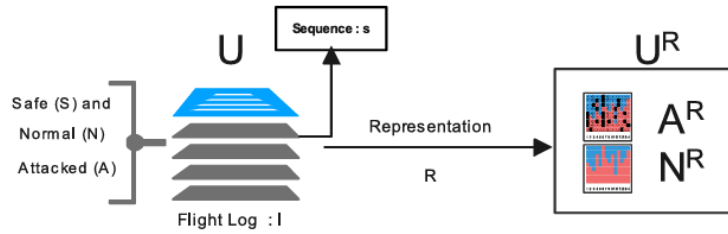


Figure 2.3: An illustration of functional attacks detection in a mapped representation of UAV flight log

E-DIDS configuration

The E-DIDS configuration enables an entire fleet of detection and reporting. The entire framework can benefit from collaborative detection and increased awareness of the overall threat landscape generated using the Central Control Module through interconnecting multiple IDS subsystems. [27] The configuration is established with security standards and appraised regularly to counteract new attacks :

(1) Identifying UAV subsystems and equipping each one with a dedicated IDS responsible for monitoring, preprocessing data, and executing the detection module.[27]

(2) Using a management module to supervise linked IDSs through configuration management, policy updates, attack identification, and coordination of detection capabilities.[27]

(3) Generalizing detection analysis and decision-making by transmitting detected attack data and logs to the central analysis module.[27]

(4) Synchronizing attack detection scores and analysis through a communication module that enables collaboration, knowledge sharing, and threat intelligence exchange between IDS subsystems.[27]

(5) Executing the report module in the Central Control Module to trigger incident alerts and automated defensive actions, such as modifying flight paths or deploying protection mechanisms.[27]

Overall, E-DIDS combines distributed IDS subsystems with centralized analysis to provide collaborative UAV attack detection and response.

Data assembly and preprocessing

The data generation and preprocessing stage involves collecting UAV data from flight logs, sensors, and system sources, cleaning missing and redundant values, encoding non-numerical features, and applying feature scaling with min–max normalization. The dataset is then labeled to distinguish normal and attack states, an additional “Alert” feature is introduced to represent IDS outputs, and the final dataset is split into training and validation sets for model training and evaluation.[27]

E-DIDS detection strategy

The IDS implemented at the E-DIDS architecture’s primer analysis and detection level serves as a detection engine based on a DL attacks detection model. Each IDS module treats the data generated by the referred linked subsystem and sends it to the central control module for performing a cooperative attack detection. The intrusions are defined at the level of the subsystems and will be transmitted as a detection analysis report with the detection rates. The DL model is in charge of learning from the data through a training phase, and validation is executed in order to evaluate the attack’s detection rate.[27]

The proposed model is based on Long Short-Term Memory (LSTM) networks, a variant of Recurrent Neural Networks designed for sequential data processing. The network processes an input sequence $[x_1, x_2, x_3, \dots, x_T]$ and produces a corresponding sequence of hidden states $[h_1, h_2, h_3, \dots, h_T]$, where each state encodes both current input information and past context. The internal mechanism is governed by standard gating equations that update the cell state and hidden representation at each time step.[27] This architecture enables the model to effectively

capture temporal dependencies in sequential data and improve classification performance on time-dependent patterns.[27]

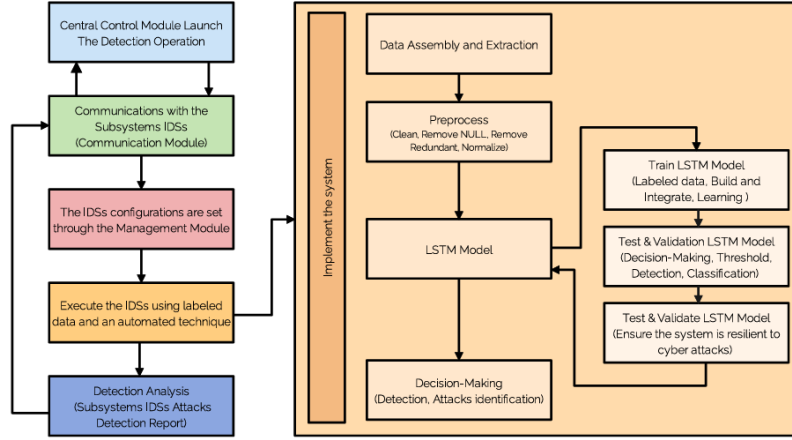


Figure 2.4: High-level flow diagram of the E-DIDS execution.

2.4 Conclusion

This chapter summarized the main categories of intrusion detection systems for UAV security, including their data collection methods, architectural designs, optimization strategies, and detection techniques. It also provided a comparative overview of representative studies in the literature. The discussion highlighted the strengths and limitations of existing approaches, which motivates the need for more efficient and distributed IDS solutions. This sets the foundation for the next chapter, where the E-DIDS architecture is introduced as the proposed framework.

Chapter 3

Chapter 3: Methodology

3.1 Introduction

This chapter presents the methodology adopted in this thesis for designing and implementing the proposed intrusion detection system for UAV environments. It begins by introducing the overall architecture and system design, followed by a detailed description of the functional modules, including the sensor, communication, management, central control, and reporting modules.

The chapter also explains the organizational structure of the UAV fleet, including cluster-based formation and cluster head selection mechanisms. It further describes the data collection process, dataset characteristics, and preprocessing pipeline used to prepare the input data for the learning model.

In addition, the proposed Kolmogorov-Arnold Network (KAN) classifier is presented along with its formulation, architectural enhancements, and inference process. The intrusion detection framework is then detailed through its subsystem decomposition, including network, navigation, and communication-based detection layers. Finally, the chapter concludes with the system workflow and the evaluation metrics used to assess performance.

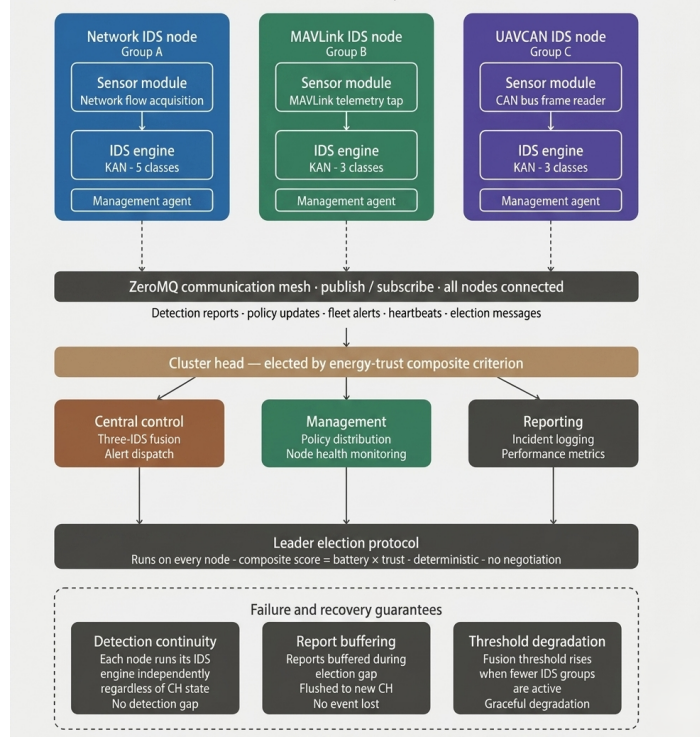


Figure 3.1: Overall System Architecture.

3.2 Adopted Architecture

This work adopts the Exhaustive Distributed Intrusion Detection System (E-DIDS) architecture presented in the related literature as the foundational security framework for protecting UAV swarm environments. The architecture is based on a distributed intrusion detection strategy in which monitoring and analysis functionalities are deployed across UAV nodes rather than relying on a centralized security entity.

The adopted architecture aims to improve scalability, resilience, and real-time threat detection capabilities within dynamic UAV networks. By distributing intrusion detection responsibilities among UAV nodes and cluster heads, the system reduces single points of failure while supporting collaborative security monitoring and coordinated anomaly detection.

Within the context of this work, the E-DIDS architecture is organized into several interconnected modules that collectively support data acquisition, communication management, distributed monitoring, intrusion detection, and security reporting operations.

In this work, the original E-DIDS architecture is adapted and extended to better align with protocol-specific intrusion detection requirements and resource-aware UAV deployment constraints. While the foundational distributed structure of E-DIDS is preserved, several key modifications are introduced. First, the detection layer is specialized into protocol-oriented subsystems, enabling dedicated analysis for network-level traffic as well as embedded communication protocols. Second, a lightweight inference engine based on a Kolmogorov-Arnold

Network (KAN) is integrated to ensure real-time detection under UAV computational limitations. Finally, the reporting and decision flow is streamlined to support faster anomaly propagation between nodes, reducing latency in collaborative detection scenarios.

Component	Original E-DIDS Architecture	Proposed Implementation (This Work)
Detection Scope	General distributed intrusion detection across UAV swarm	Specialized multi-layer detection: Network IDS, MAVLink-based IDS, and Embedded Communication IDS
Detection Model	General machine learning-based detection without protocol specialization	Lightweight Kolmogorov-Arnold Network (KAN)-based inference engine optimized for real-time UAV constraints
System Structure	Distributed architecture with cluster-based coordination	Preserved distributed architecture with enhanced protocol-aware subsystem decomposition
Processing Strategy	Partially centralized processing at cluster or coordination nodes	Fully distributed inference executed at UAV node level (edge-based processing)
Protocol Awareness	Protocol-agnostic intrusion detection design	Protocol-aware detection tailored to UAV communication layers (network, MAVLink, UAVCAN)
Resource Optimization	General scalability considerations	Explicit lightweight design optimized for onboard UAV execution and low latency inference

Table 3.1: Comparison between the original E-DIDS architecture and the proposed implementation.

3.3 System Modules

The architecture is composed of multiple interconnected modules, each responsible for a specific function within the distributed intrusion detection process. This modular organization enables efficient sensing, detection, communication, coordination, and management across the UAV swarm.

The following subsections present the main system modules and their respective roles within the proposed architecture :

3.3.1 Sensor Module

The Sensor Module represents the data acquisition and preprocessing component of the adopted E-DIDS architecture. It is responsible for collecting raw UAV operational data and transforming it into structured feature representations suitable for intrusion detection analysis.

This module is strictly divided into three sequential layers:

Layer 1: Data Acquisition Layer

This layer collects raw input data from UAV-related sources. In a real deployment scenario, this layer interfaces directly with UAV communication channels such as network interfaces and MAVLink serial links to capture real-time telemetry and packet flows.

Layer 2: Preprocessing Layer

This layer applies the exact transformation pipeline used during offline training. Importantly, no re-fitting occurs at inference time, ensuring strict prevention of data leakage and maintaining model consistency between training and deployment environments.

Layer 3: Feature Output Layer

The resulting processed feature vector is pushed into a thread-safe queue, which acts as the input buffer for the IDS inference engine. The Sensor Module does not perform classification, its role is strictly limited to producing model-ready feature vectors.

This design enables heterogeneous sensing across the UAV fleet while maintaining a unified preprocessing pipeline.

3.3.2 Communication Module

The Communication Module is responsible for managing all inter-node and intra-node information exchange within the UAV swarm, ensuring continuous dissemination of detection results, control signals, and coordination messages across the system.

It is structured into two logical layers:

Layer 1: Transport Layer

Provides the underlying mechanism for message exchange between UAV nodes, ensuring reliable and low-latency delivery of information across the swarm to maintain connectivity and coordination.

Layer 2 : Nucleus Layer

Defines the structure and meaning of exchanged information, where each message carries a semantic payload such as IDS detection reports, fleet-wide alerts, policy updates, and coordination signals, enabling consistent interpretation across heterogeneous nodes.

A lightweight buffering mechanism is also employed to temporarily store outgoing messages when immediate delivery is not possible, ensuring continuity of communication and preventing information loss.

3.3.3 Management Module

The Management Module represents the control and configuration layer of the distributed intrusion detection architecture. Its role is to regulate system-wide operational behavior through adaptive policies, maintain awareness of node health, and ensure coordinated execution of detection and communication processes across the system. Unlike detection components, this module does not perform classification; instead, it defines and enforces the operational rules under which all other components operate.

To ensure modularity and clarity, the Management Module is structured into two functional layers: the Configuration Layer and the Monitoring Layer.

Configuration Layer

The Configuration Layer defines the global operational policy governing the behavior of the system. This policy is represented as a structured parameter set:

$$\mathcal{P} = \{\theta_1, \theta_2, \theta_3, \phi_3, \phi_2, \phi_1, \tau_h, \tau_s, \tau_{sample}, \tau_a, \tau_{silent}\}$$

where:

- $\theta_1, \theta_2, \theta_3$ represent decision thresholds used by the underlying detection components;
- ϕ_3, ϕ_2, ϕ_1 represent fusion decision thresholds depending on the number of active decision sources contributing to a final output;
- τ_h defines the heartbeat interval used for periodic liveness signaling;
- τ_s defines the silence timeout used for failure detection and inactivity estimation;
- τ_{sample} controls the sampling rate of data acquisition modules;
- τ_a defines the cooldown period between consecutive system-wide alerts;
- τ_{silent} defines the threshold used to classify a node as non-responsive.

These parameters define a global control policy that governs the behavior of all system components introduced in subsequent sections.

Policy updates are distributed across all nodes using a version-controlled mechanism. Let $\mathcal{P}^{(v)}$ denote the policy at version v . A new configuration is accepted only if:

$$v_{new} > v_{current}$$

This ensures consistent and monotonic evolution of system behavior while preventing outdated configurations from overriding newer ones.

Monitoring Layer

The Monitoring Layer is responsible for maintaining real-time awareness of system-wide node status. It tracks node activity and detects communication degradation based on temporal message behavior.

For each node i , a state descriptor is maintained:

$$\mathcal{S}_i = (t_i^{last}, r_i, \kappa_i)$$

where t_i^{last} represents the timestamp of the most recent received message, r_i represents the number of reports received, and κ_i represents the set of active processing components associated with the node.

The inactivity duration of a node is defined as:

$$\Delta t_i = t_{current} - t_i^{last}$$

A node is classified as operational if:

$$\Delta t_i < \tau_{silent}$$

and classified as silent otherwise:

$$\Delta t_i \geq \tau_{silent}$$

Nodes identified as silent are reported for system-level awareness and logging purposes, without directly affecting decision-making processes.

The monitoring process produces a global system health map:

$$\mathcal{H} = \{(i, status_i) \mid i \in \mathcal{N}\}$$

where each node is labeled as either OK or SILENT based on its observed activity.

Functional Overview

Overall, the Management Module ensures:

- consistent enforcement of global system policies across all components;
- adaptive configuration of detection sensitivity and operational parameters;
- continuous monitoring of node availability and responsiveness;
- version-controlled dissemination of configuration updates across the system.

This layered design allows the Management Module to operate as a lightweight yet essential control mechanism ensuring stability and coherence in the distributed intrusion detection architecture.

3.3.4 Central Control Module

The Central Control Module represents the decision fusion core of the distributed intrusion detection architecture. Its primary role is to transform multiple heterogeneous detection reports generated across the system into a single coherent fleet-level security decision. Unlike local detection components, this module does not perform feature extraction or classification; instead, it operates on aggregated inference outputs and enforces a unified decision logic that governs system-wide response behavior.

To ensure a structured decision pipeline, the Central Control Module is organized into three functional layers: the Analysis Layer, the Decision Layer, and the Response Layer.

Analysis Layer

The Analysis Layer is responsible for collecting and organizing detection reports received from distributed nodes within a temporal observation window. Each report is represented as:

$$r_i = (\hat{y}_i, c_i, t_i, s_i)$$

where $\hat{y}_i$ denotes the predicted class label, $c_i \in [0, 1]$ is the associated confidence score, t_i is the timestamp of the report, and s_i identifies the source node.

To ensure temporal relevance, only recent reports within a sliding window ΔT are considered:

$$\mathcal{R}(t) = \{r_i \mid t - t_i \leq \Delta T\}$$

These reports are then grouped according to their originating detection stream, allowing the system to maintain separate statistical views of concurrent detection sources while preserving temporal coherence.

For each group g , the mean confidence is computed as:

$$\mu_g = \frac{1}{|\mathcal{R}_g|} \sum_{r \in \mathcal{R}_g} c_r$$

where $\mathcal{R}_g$ represents the subset of reports belonging to group g .

Decision Layer

The Decision Layer performs dynamic fusion of the aggregated confidence values. Let $G_a \subseteq G$ denote the set of active detection groups contributing non-empty reports at time t . The number of active groups is defined as:

$$n_a = |G_a|$$

The fused confidence score is computed adaptively depending on the level of redundancy in the available evidence:

$$C_f = \begin{cases} \frac{1}{3} \sum_{g \in G_a} \mu_g & \text{if } n_a = 3 \\ \frac{1}{2} \sum_{g \in G_a} \mu_g & \text{if } n_a = 2 \\ \mu_g & \text{if } n_a = 1 \end{cases}$$

This formulation ensures that stronger consensus across independent detection sources increases confidence stability while reducing variance caused by isolated false positives.

A decision is triggered when the fused confidence exceeds a mode-dependent threshold:

$$C_f \geq \theta(n_a)$$

where the threshold function $\theta(\cdot)$ is defined as:

$$\theta(n_a) = \begin{cases} \theta_3 & \text{if } n_a = 3 \\ \theta_2 & \text{if } n_a = 2 \\ \theta_1 & \text{if } n_a = 1 \end{cases}$$

This adaptive thresholding mechanism ensures that the system becomes more conservative when fewer independent detection sources are available, and more permissive when multi-source corroboration is present.

Response Layer

Once a decision is triggered, the Response Layer is responsible for executing system-wide actions. To prevent repeated alerts caused by redundant detections, a cooldown constraint is

enforced:

$$t - t_{last} \geq \tau_{cooldown}$$

where t_{last} is the timestamp of the most recent alert and $\tau_{cooldown}$ is a policy-controlled suppression interval.

When activated, the module generates a structured threat descriptor:

$$\mathcal{T} = (C_f, \theta, n_a, \kappa_g, t)$$

where κ_g denotes the majority class inferred per detection group.

The threat descriptor is then broadcast to all nodes in the system, ensuring synchronized awareness of the detected anomaly. In parallel, the event is logged for auditability and post-analysis.

In certain high-confidence conditions or when internal compromise signals are detected, the system escalates the response level to immediate containment mode; otherwise, a monitoring directive is issued.

Functional Overview

Overall, the Central Control Module provides a structured mechanism for:

- aggregating distributed detection outputs into a unified representation;
- performing adaptive multi-source confidence fusion;
- dynamically adjusting decision sensitivity based on available evidence;
- coordinating real-time system-wide response actions.

This design ensures that the distributed nature of the detection system is preserved while enabling centralized, consistent, and robust security decision-making at the fleet level.

3.3.5 Reporting Module

The Reporting Module serves as the monitoring and assessment component of the E-DIDS architecture. Its primary role is to maintain a comprehensive record of system activities and security events, providing the information required to evaluate the effectiveness and reliability of the distributed intrusion detection framework.

The module is organized into two functional layers:

Layer 1 : Data Aggregation Layer

Collects and consolidates information generated by the various system components, including detection outcomes, security incidents, coordination events, and operational activities occurring throughout the UAV fleet.

Layer 2 : Response Layer

Analyzes the accumulated information to assess the behavior and performance of the system, supporting the measurement of detection effectiveness, fleet status, and overall security performance.

By maintaining a unified view of system events and outcomes, the Reporting Module supports system monitoring, performance validation, and post-mission analysis, providing the evidence required to evaluate the proposed E-DIDS architecture.

3.4 Fleet Organization

The UAV swarm is organized as a distributed set of cooperative nodes that work together to achieve continuous intrusion detection across the operational environment. Instead of acting as isolated units, all UAVs contribute to a shared detection objective through coordinated information exchange and collective awareness.

The fleet is composed of specialized UAV types, each acting as an expert in a specific observation domain. Some nodes focus on network-level behavior, while others specialize in Navigation-related and Intra-Vehicle activity . This specialization allows the system to capture diverse perspectives of the same operational environment, improving overall detection coverage and robustness.

Although each UAV operates based on its own observations, all nodes follow a unified operational logic that enables consistent interpretation of detected events. This ensures that individual detections can be meaningfully combined at higher system levels for global decision-making.

Overall, the fleet is structured as a set of complementary expert nodes that jointly provide distributed intelligence, enabling accurate and scalable intrusion detection across the UAV swarm.

3.5 Cluster-Based Organization

The fleet follows a cluster-based organization in which one UAV is designated as the Cluster Head (CH), while all remaining UAVs operate as follower nodes. Follower nodes perform

local sensing and intrusion detection, generating security reports based on their assigned detection capabilities. These reports are forwarded to the Cluster Head, which acts as the coordination and decision-making entity of the fleet. By maintaining a global view of the swarm, the Cluster Head aggregates detection information from multiple UAVs, performs collaborative threat assessment, disseminates operational policies, and broadcasts security alerts when necessary. This organization combines distributed detection with centralized coordination, enabling efficient fleet-wide situational awareness while preserving the autonomous operation of individual UAVs.

3.5.1 Cluster Head

The Cluster Head is the elected coordination authority of the fleet. It is the only node that runs the three higher-order coordination modules: the Central Control Module, the Management Module, and the Reporting Module. No follower node performs these functions. The Cluster Head's designation is dynamic, it is earned through election and can transfer to another node at any time if the current holder fails or loses eligibility.

Beyond coordination, the Cluster Head also carries its own local detection pipeline like any other node. Its detection output is injected into the Central Control Module alongside reports from other fleet nodes, ensuring that fleet-level decisions benefit from both local and distributed observations.

Within the Cluster Head, the coordination modules fulfill the following roles:

- **Management Module:** Responsible for maintaining and enforcing the global operational policy of the fleet. It ensures that all nodes operate under consistent configuration parameters and provides system-wide control over operational constraints and security settings.
- **Central Control Module:** Responsible for aggregating detection reports from all fleet nodes and producing a unified fleet-level threat assessment. It performs multi-source fusion of incoming alerts and determines whether a coordinated response is required at the fleet level.
- **Reporting Module:** Responsible for maintaining the global incident log. It records confirmed security events and ensures traceability of all fleet-level detection and response actions.

3.5.2 Cluster Head Election

The coordination role within the fleet is assigned through a distributed election mechanism designed to ensure both operational efficiency and security resilience. Rather than relying on a single selection criterion, the protocol combines energy availability with behavioral reliability, allowing the system to avoid delegating coordination authority to nodes that are either resource-constrained or behaviorally unreliable.

Each node maintains a local trust indicator that reflects its historical consistency with the fleet's collective detection outcomes. This indicator evolves gradually over time based on whether the node's reported alerts align with or diverge from globally validated decisions. As a result, nodes that consistently produce reliable detections gradually increase their trust level, while those exhibiting inconsistent or misleading behavior experience a progressive reduction in trust.

At any time t , each node i is characterized by its residual energy $\beta_i(t)$ and trust score $\tau_i(t)$. These two variables jointly define the node's eligibility for coordination through a composite score defined as:

$$S_i(t) = \frac{\beta_i(t)}{100} \cdot \tau_i(t)$$

This multiplicative formulation ensures that both energy and trust contribute simultaneously to the election outcome. A node with high energy but low trust, or vice versa, cannot dominate the selection process, thereby enforcing a balanced and security-aware evaluation.

The node selected to assume coordination is the one that maximizes this composite score over all known fleet members:

$$i^* = \arg \max_{i \in \mathcal{N}} S_i(t)$$

To guarantee deterministic behavior across all nodes, ties are resolved using a fixed ordering based on node identifiers. This ensures that every node independently arrives at the same election outcome without requiring additional communication rounds.

The trust score $\tau_i(t)$ is updated dynamically based on feedback from the global fusion process. After each evaluation cycle, it is adjusted using a smoothing mechanism:

$$\tau_i(t+1) = \alpha \cdot s(t) + (1 - \alpha)\tau_i(t)$$

where α is a smoothing factor and $s(t)$ represents the outcome signal derived from system-

level validation.

To maintain stability, the trust value is constrained within a bounded interval:

$$\tau_i(t) \in [0.05, 1.0]$$

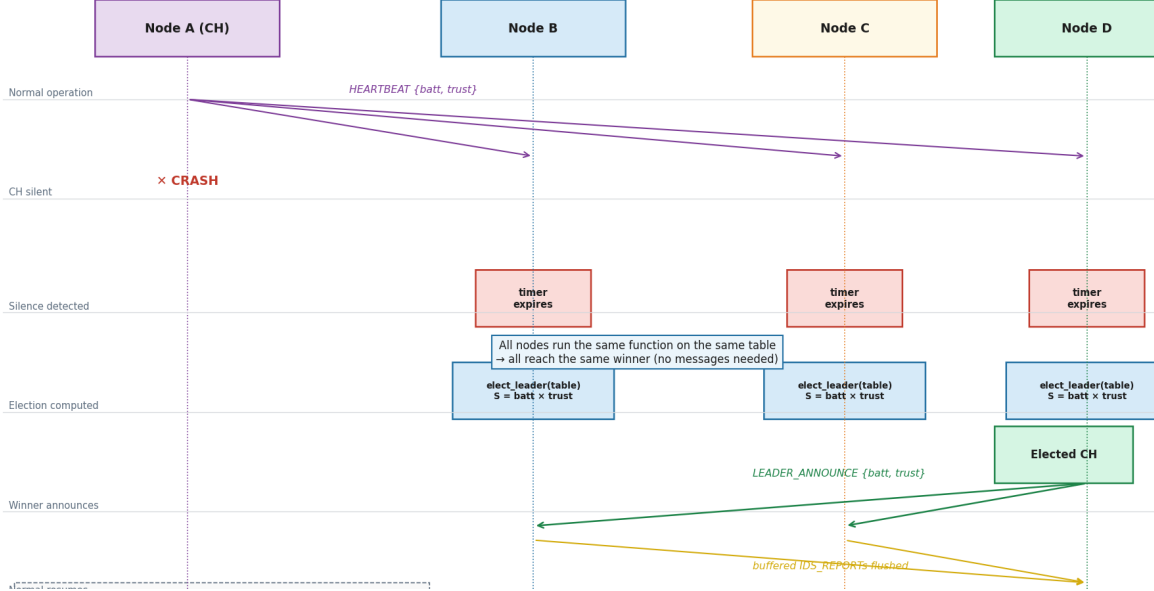


Figure 3.2: Cluster Head Election Protocol

Overall, this mechanism transforms coordination selection into a fully distributed optimization process based on energy and behavioral credibility. It ensures continuity of leadership while preventing unreliable or compromised nodes from gaining control, even when they possess sufficient energy resources.

3.6 Data Collection and Preprocessing

3.6.1 Datasets Description

This study utilizes three publicly available datasets that cover different layers of UAV communication and security. Each dataset corresponds to a specific intrusion detection module within the proposed architecture, enabling comprehensive coverage of UAV network, protocol, and intra-system attack surfaces.

UAVIDS2025 Dataset (Network Intrusion Detection)

The UAVIDS2025 dataset is obtained from Kaggle [28]. It is used to train and evaluate the network-level intrusion detection module.

This dataset contains UAV network traffic records representing both normal communication and multiple types of cyberattacks. Each instance consists of flow-based features extracted from packet-level statistics, including temporal, statistical, and behavioral characteristics of network communication.

The dataset is designed for multi-class classification, where each sample is labeled according to its corresponding traffic category. It serves as the primary benchmark dataset for evaluating the performance of the proposed KAN-based Network IDS.

MAVLink Intrusion Dataset

The MAVLink intrusion dataset is obtained from IEEE DataPort [29]. It focuses on security threats targeting the MAVLink communication protocol, which is widely used in UAV command-and-control systems.

The dataset includes telemetry and command exchange logs under both normal flight conditions and adversarial scenarios. Attack types include message injection, command spoofing, and telemetry manipulation.

Due to its original structure consisting of multiple fragmented files and inconsistent labeling formats, the dataset was preprocessed and consolidated into a unified CSV file. The resulting dataset contains clearly defined normal and attack classes and is used for evaluating the MAVLink intrusion detection module.

UAVCAN Intrusion Dataset

The UAVCAN intrusion dataset is obtained from IEEE DataPort [30]. It targets the UAVCAN protocol, which is used for low-level intra-UAV communication and distributed control systems.

This dataset includes normal operational traffic as well as malicious behaviors such as node impersonation, message tampering, and bus-level anomalies. These attacks reflect threats at the embedded communication layer of UAV systems.

Similar to the MAVLink dataset, the original dataset was provided in a distributed format and required preprocessing and aggregation into a single structured CSV file. The final dataset includes labeled normal and attack samples and is used to evaluate the UAVCAN intrusion detection module.

3.6.2 Preprocessing Pipeline

The preprocessing stage defines a unified transformation pipeline applied across all datasets to ensure consistency, comparability, and reproducibility of the intrusion detection models. The pipeline follows a structured sequence consisting of cleaning, encoding, normalization, dataset partitioning, feature selection, and class imbalance handling.

1. Data Cleaning and Normalization Raw datasets are first cleaned by removing identifier-based and non-informative columns that do not contribute to learning. Missing and infinite values are eliminated to ensure numerical stability during training. All remaining categorical attributes are encoded into numerical representations using label encoding. This step ensures compatibility with downstream machine learning models.

Feature scaling is then applied using standardization, which transforms features to zero mean and unit variance. This prevents dominant feature bias and stabilizes gradient-based and distance-sensitive learning algorithms.

2. Dataset Splitting Strategy The dataset is first partitioned into training, validation, and testing subsets using a stratified strategy to preserve class distribution. This ensures that all attack categories remain proportionally represented across the different subsets, preventing evaluation bias and ensuring reliable performance estimation.

3. Feature Selection using RFE (Training-Only) Recursive Feature Elimination (RFE) is applied exclusively on the training set after dataset partitioning. This design ensures that feature selection is fully independent from validation and test data, thereby preventing any form of information leakage during preprocessing.

RFE is a wrapper-based feature selection method that recursively removes the least important features based on model performance until an optimal subset is obtained [31].

Unlike filter-based methods such as chi-square or mutual information, RFE evaluates feature importance in the context of a predictive model, allowing it to capture feature interactions rather than relying on independent statistical scoring. This makes it particularly suitable for intrusion detection problems, where dependencies between network features are often non-linear and highly correlated [31, 32].

Filter methods, while computationally efficient, often fail to account for redundancy between features, which may result in suboptimal feature subsets in high-dimensional network traffic environments.

Once the optimal feature subset is selected on the training data, the same feature mask is applied consistently to the validation and test sets without re-fitting.

4. Handling Class Imbalance using SMOTE To address class imbalance, the Synthetic Minority Over-sampling Technique (SMOTE) is applied exclusively to the training set after feature selection. SMOTE generates synthetic samples for minority classes by interpolating between existing samples rather than duplicating them [33].

This approach is preferred over random oversampling because it reduces the risk of overfitting. Random oversampling simply replicates minority samples, which may lead to memorization rather than generalization. In contrast, SMOTE introduces variability in synthetic samples, improving decision boundary smoothness and classifier robustness [33, 34].

Importantly, SMOTE is never applied to validation or test sets to ensure unbiased performance evaluation and prevent data leakage.

5. Output Artefacts The preprocessing pipeline produces standardized transformation artifacts including the fitted scaler, label encoder, and selected feature mask. These artifacts are reused during inference to guarantee consistency between training and deployment environments, ensuring that no re-fitting occurs during runtime.

3.7 Kolmogorov-Arnold Network (KAN) Classifier

The intrusion detection system is built upon a Kolmogorov-Arnold Network (KAN)-inspired classifier, originally introduced as a neural architecture that replaces conventional node-wise activation functions with learnable univariate functions defined on network edges [9]. Unlike traditional multilayer perceptrons, which apply fixed nonlinearities at neuron level, KAN decomposes multivariate functions into compositions of adaptive one-dimensional functions, offering improved flexibility in function approximation.

Given an input feature vector $X \in \mathbb{R}^d$, the model constructs nonlinear transformations using a combination of spline-based function approximation and residual activation functions. Each edge-wise transformation is defined as:

$$\phi_{ij}(x_i) = \sum_{m=1}^M c_{ijm} B_m(x_i) + w_{ij} \cdot \sigma(x_i)$$

where:

- $B_m(\cdot)$ denotes B-spline basis functions [35],
- c_{ijm} are learnable spline coefficients,
- w_{ij} is a learnable weight parameter,
- $\sigma(\cdot)$ is the SiLU activation function [36],
- M is the number of spline basis functions.

The output of each KAN layer is computed as:

$$h_j = \sum_{i=1}^d \phi_{ij}(x_i)$$

This formulation enables the model to capture both smooth global dependencies (via splines) and local nonlinear variations (via SiLU-based residual terms).

3.7.1 Adaptive Grid Construction

To improve numerical stability on heterogeneous UAV telemetry distributions, the spline grid is constructed using percentile-based adaptive bounds:

$$g_{\min} = P_1(X), \quad g_{\max} = P_{99}(X)$$

This adaptive formulation ensures that spline basis functions are concentrated within the most informative regions of the feature space, reducing sensitivity to extreme outliers.

3.7.2 Architectural Enhancements

To improve training stability and generalization in real-time intrusion detection scenarios, several modifications are introduced:

- **Layer Normalization** applied after each hidden layer to stabilize intermediate feature distributions [37].
- **Label Smoothing** used during optimization to reduce overconfidence on noisy or synthetic samples [38].
- **Glorot Initialization** applied to spline and residual weights to preserve variance across layers [39].
- **Gradient Clipping** applied to prevent gradient explosion during backpropagation.

These modifications transform the original KAN formulation into a more robust architecture suitable for deployment in constrained and noisy environments such as UAV networks.

3.7.3 Classification Objective

Let $\hat{y} \in \mathbb{R}^C$ denote the predicted class probability distribution over C intrusion classes. The model is trained using a label-smoothed cross-entropy loss:

$$\mathcal{L} = - \sum_{i=1}^N \sum_{c=1}^C \tilde{y}_{ic} \log(\hat{y}_{ic})$$

where the smoothed target distribution is defined as:

$$\tilde{y}_{ic} = \begin{cases} 1 - \varepsilon & \text{if } c = y_i \\ \frac{\varepsilon}{C-1} & \text{otherwise} \end{cases}$$

3.7.4 Inference

The final prediction is obtained through a softmax transformation:

$$P(y = c \mid X) = \frac{\exp(z_c)}{\sum_{k=1}^C \exp(z_k)}$$

$$\hat{y} = \arg \max_c P(y = c \mid X)$$

3.7.5 Summary

This KAN-inspired classifier extends the original Kolmogorov-Arnold framework [9] with stability-oriented and deployment-driven modifications. The resulting architecture maintains the expressive power of functional decomposition while improving robustness for real-time UAV intrusion detection systems.

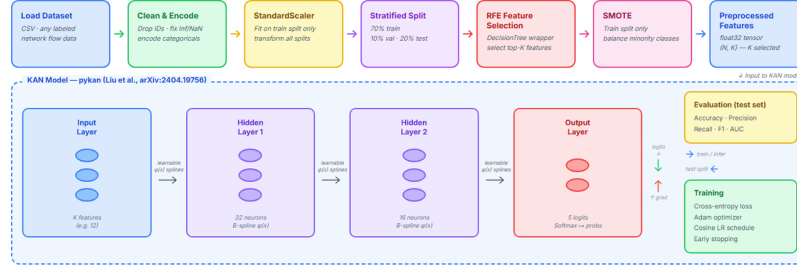


Figure 3.3: Data Preprocessing Pipeline and KAN Model Architecture .

3.8 Intrusion Detection Framework

This section presents the proposed Intrusion Detection Framework designed for UAV swarm environments. The framework is structured as a distributed, multi-layer detection system that operates across heterogeneous communication protocols. It is built on top of a shared learning engine based on Kolmogorov-Arnold Networks (KAN), which provides the core classification capability across all subsystems.

The architecture follows a modular design in which each intrusion detection subsystem is responsible for monitoring a specific communication domain, while a central IDS engine performs global aggregation and decision fusion.

3.9 Intrusion Detection Subsystems

This work proposes a multi-layer intrusion detection framework composed of three specialized subsystems. Each subsystem targets a distinct UAV communication layer while sharing a unified learning backbone based on the Kolmogorov-Arnold Network (KAN). This design enables protocol-aware anomaly detection while maintaining computational efficiency suitable for embedded UAV environments.

3.9.1 Network Intrusion Detection System

The Network Intrusion Detection System operates at the general UAV communication layer, where heterogeneous traffic flows are analyzed to detect malicious behavior. The system is

trained using the UAVIDS2025 dataset, which contains both normal and attack traffic samples.

Input Representation Each network traffic sample is initially represented as a high-dimensional feature vector extracted from raw flow data. After preprocessing and stratified dataset splitting, Recursive Feature Elimination (RFE) is applied exclusively on the training set to select the most informative features.

The final reduced representation is:

$$\mathbf{x} \in \mathbb{R}^{12} \quad (3.1)$$

The same selected feature subset is consistently applied to validation and test sets to ensure fair evaluation and to avoid data leakage.

Model Architecture The classification model follows a KAN-based structure:

$$12 \rightarrow 64 \rightarrow 32 \rightarrow C \quad (3.2)$$

where C denotes the number of output classes.

3.9.2 Navigation and Telemetry Intrusion Detection System

The Navigation and Telemetry Intrusion Detection System targets the MAVLink communication protocol [40], which is widely used in UAV systems for telemetry exchange, navigation control, and mission commands.

MAVLink is a lightweight message-based protocol designed for constrained aerial systems, enabling efficient communication between flight controllers and ground control stations. It defines structured message types for position reporting, sensor feedback, and command execution.

Within this framework, the system monitors navigation messages and telemetry streams to detect anomalies such as malicious command injection, spoofed telemetry data, abnormal flight parameter updates, and deviations in expected message sequences.

Input Representation Each MAVLink telemetry sample is represented as a structured feature vector derived from navigation and control messages. After preprocessing and stratified dataset splitting, Recursive Feature Elimination (RFE) is applied exclusively on the training set to select the most relevant features.

The final representation is:

$$\mathbf{x} \in \mathbb{R}^{10} \quad (3.3)$$

The same feature subset is applied to validation and test sets to ensure consistency and avoid data leakage.

3.9.3 Embedded Communication Intrusion Detection System

The Embedded Communication Intrusion Detection System operates over the UAVCAN protocol (Cyphal standard) [41], which is designed for deterministic and reliable embedded communication in avionics and robotic systems.

UAVCAN enables real-time communication between onboard components such as sensors, actuators, and embedded controllers. It is widely used in safety-critical UAV subsystems due to its structured message passing and fault-tolerant design.

The system focuses on detecting anomalies at the embedded communication layer, including sensor spoofing, actuator manipulation, and irregular node-level communication behavior.

Input Representation Each embedded communication sample is initially represented in a high-dimensional space derived from UAVCAN message features. After preprocessing and stratified dataset splitting, Recursive Feature Elimination (RFE) is applied exclusively on the training set to identify the most discriminative features.

The resulting compact representation is:

$$\mathbf{x} \in \mathbb{R}^8 \quad (3.4)$$

The same selected feature subset is then applied to validation and test sets to ensure consistency and prevent data leakage during evaluation.

Model Architecture The classifier is defined as:

$$8 \rightarrow 64 \rightarrow 32 \rightarrow C \quad (3.5)$$

IDS	Category	Monitored Elements	Dataset	Attacks Detected
Network IDS	Network-based (UAV traffic layer)	Packets, flows, routing behavior, telemetry streams	UAVIDS2025 dataset [28]	DoS, spoofing, jamming, generic anomalies
Navigation and Telemetry IDS	Protocol-based (MAVLink layer)	Navigation messages, telemetry data, command sequences	UAV MAVLink dataset [29]	Command injection, GPS spoofing, parameter tampering
Embedded Communication IDS	Embedded layer (UAVCAN / Cyphal)	Sensor signals, actuator commands, node communication	UAVCAN dataset [30]	Sensor spoofing, actuator manipulation, node anomalies

Table 3.2: Summary of intrusion detection subsystems used in the proposed E-DIDS architecture.

3.9.4 Intrusion Detection Engine (Runtime Inference Layer)

To operationalize the proposed intrusion detection subsystems, an inference engine is deployed at the system level to enable real-time decision-making within UAV environments. This component, referred to as the Intrusion Detection Engine, acts as the execution layer that connects feature acquisition modules to the trained KAN-based IDS models.

Unlike the training phase, which focuses on optimization and feature learning, the IDS engine is responsible for lightweight deployment, streaming inference, and event-driven anomaly reporting under strict computational constraints.

System Role and Integration

Each IDS subsystem (Network, Navigation and Telemetry, and Embedded Communication) is deployed as an independent inference instance within the engine. The engine loads the corresponding trained KAN model and associates it with its respective communication layer.

Let the incoming feature stream be denoted as:

$$\mathbf{x}_t \in \mathbb{R}^d \quad (3.6)$$

where $d \in \{12, 10, 8\}$ depending on the IDS subsystem.

Each subsystem operates independently, but shares the same inference logic and decision policy.

KAN-Based Inference Process

At runtime, each feature vector is processed through the trained Kolmogorov–Arnold Network model under a no-gradient evaluation mode:

$$\hat{\mathbf{y}}_t = \text{KAN}(\mathbf{x}_t) \quad (3.7)$$

The posterior probability distribution is obtained using the softmax function:

$$P(y_i|\mathbf{x}_t) = \frac{e^{z_i}}{\sum_{j=1}^C e^{z_j}} \quad (3.8)$$

where z_i represents the output logits of the KAN model.

The predicted class is determined by:

$$\hat{c}_t = \arg \max_i P(y_i|\mathbf{x}_t) \quad (3.9)$$

Decision Logic and Anomaly Triggering

The IDS engine applies a confidence-based decision mechanism to distinguish between normal and malicious behavior. A detection is flagged as an anomaly if:

$$P(\hat{c}_t|\mathbf{x}_t) \geq \tau \quad \wedge \quad \hat{c}_t \notin \mathcal{N} \quad (3.10)$$

where:

- τ is the detection threshold,
- $\mathcal{N}$ represents the set of normal traffic classes,
- $P(\hat{c}_t|\mathbf{x}_t)$ is the model confidence.

This mechanism ensures that only high-confidence non-normal predictions are classified as attacks, reducing false positives in real-time UAV environments.

Streaming Architecture and Concurrency Model

The IDS engine is implemented as a multithreaded streaming system. Feature vectors are received through an asynchronous queue and processed sequentially in a dedicated inference thread.

Let $\mathcal{Q}_f$ denote the feature queue and $\mathcal{Q}_r$ the reporting queue. The system operates as:

$$\mathcal{Q}_f \rightarrow \text{IDS Engine} \rightarrow \mathcal{Q}_r \quad (3.11)$$

This design decouples feature extraction from inference, ensuring non-blocking execution and stable throughput under high-frequency UAV telemetry streams.

Operational Statistics and Monitoring

The engine maintains runtime statistics for system monitoring, including:

- Total number of inferences:

$$N_{\text{inf}} = \sum_t 1 \quad (3.12)$$

- Number of detected alerts:

$$N_{\text{alert}} = \sum_t \mathbb{I}(\text{attack}_t) \quad (3.13)$$

- Inference rate:

$$R = \frac{N_{\text{inf}}}{T} \quad (3.14)$$

where T represents system uptime.

These metrics allow runtime evaluation of system responsiveness and anomaly density across UAV missions.

3.10 Overall System Workflow

The system architecture operates as a fully distributed, event-driven security framework deployed across a fleet of unmanned aerial vehicles. Its operation is structured as a continuous lifecycle composed of four main phases: initialization, local detection, collaborative escalation, and resilience recovery. These phases are executed cyclically throughout mission execution.

Initialization phase. At deployment, each UAV node initializes its local sensing, pre-processing, and communication modules, and joins the UAV mesh network. In parallel, each node loads its pre-trained intrusion detection model based on a Kolmogorov–Arnold Network (KAN), along with the associated feature selection and normalization parameters learned during the offline training stage.

Once all nodes are operational, a distributed leader election process is executed to select the Cluster Head based on residual energy and system-defined selection criteria. The elected Cluster Head is responsible for coordinating global information exchange and aggregating security decisions, while all other UAVs operate as independent detection agents.

Local detection phase. Each UAV continuously performs local intrusion detection on its respective data streams, depending on its assigned communication domain (e.g., MAVLink, Network traffic, or UAVCAN bus data). Incoming raw data is processed through a fixed preprocessing pipeline, including feature engineering, normalization, and Recursive Feature Elimination (RFE)-based feature selection, producing compact feature vectors.

These feature vectors are then fed into the local KAN classifier, which performs nonlinear spline-based transformations and learns complex feature interactions through adaptive basis functions. The model outputs a probability distribution over the predefined attack and normal classes of the corresponding subsystem.

A detection event is triggered when the confidence of any attack class exceeds a predefined decision threshold, ensuring fully decentralized and real-time intrusion detection without reliance on centralized computation.

Collaborative escalation phase. When a detection event is triggered, the UAV generates a structured alert containing the predicted class, confidence score, and contextual metadata. This alert is transmitted to the Cluster Head through the UAV communication mesh.

The Cluster Head aggregates alerts within a sliding time window, grouping them by subsystem and source node. A global decision is then computed by fusing multiple KAN-based local predictions across the fleet, enabling cross-validation of suspicious behavior and reducing false positives.

When the aggregated confidence exceeds a system-level threshold, a global alert is issued and broadcast to all UAV nodes, triggering coordinated defensive actions if required.

Resilience and recovery phase. The system ensures operational continuity through full decentralization of local intrusion detection. Each UAV continues executing its KAN-based detection pipeline independently, even in the absence of a Cluster Head or during temporary communication disruptions.

In the event of Cluster Head failure, nodes detect the loss of coordination within a bounded timeout and initiate a new leader election process. During this transition, locally generated alerts are buffered and automatically forwarded once a new Cluster Head is established, ensuring no loss of security-critical information.

This design guarantees continuous and resilient intrusion detection across dynamic UAV network conditions, while preserving scalability, fault tolerance, and consistent decision-making across heterogeneous communication domains.



Figure 3.4: System Workflow.

3.11 Evaluation Metrics

The evaluation of the proposed UAV intrusion detection system is conducted along two complementary dimensions: (i) the classification performance of the Kolmogorov-Arnold Network (KAN)-based IDS, and (ii) the operational performance of the distributed UAV security architecture. The use of multiple evaluation metrics ensures a comprehensive assessment of both detection accuracy and system robustness [42].

3.11.1 Classification Performance Metrics

To evaluate the effectiveness of the proposed model, standard classification metrics are employed. These metrics are particularly important in intrusion detection tasks due to the inherent imbalance between normal and attack traffic classes [43].

Accuracy measures the proportion of correctly classified samples:

$$Accuracy = \frac{\sum_{c=0}^{C-1} TP_c}{N} \quad (3.15)$$

where TP_c denotes true positives for class c , C is the number of classes, and N is the total number of samples.

However, accuracy alone is insufficient for imbalanced datasets, as it may be biased toward majority classes.

Precision is defined as:

$$P_c = \frac{TP_c}{TP_c + FP_c} \quad (3.16)$$

It measures the proportion of correctly identified positive predictions among all predicted positives.

Recall is defined as:

$$R_c = \frac{TP_c}{TP_c + FN_c} \quad (3.17)$$

It measures the ability of the model to detect all actual positive samples. **F1-score** is the harmonic mean of precision and recall:

$$F1_c = \frac{2 \cdot P_c \cdot R_c}{P_c + R_c} \quad (3.18)$$

The macro-averaged F1-score is computed as:

$$F1_{macro} = \frac{1}{C} \sum_{c=0}^{C-1} F1_c \quad (3.19)$$

This metric ensures equal contribution from all classes, making it suitable for intrusion detection systems where minority attack classes are critical.

Confusion Matrix provides a complete representation of classification performance, where each element M_{ij} represents the number of samples from class i predicted as class j .

ROC-AUC evaluates the discriminative ability of the classifier across all decision thresholds using a one-vs-rest strategy. The macro-average ROC-AUC is computed across all classes [44].

3.11.2 Operational Performance Metrics

Beyond classification performance, the system is evaluated in terms of distributed execution efficiency and robustness.

Detection Latency measures the time between input acquisition and classification output:

$$\Delta t_{detect} = t_{result} - t_{sample} \quad (3.20)$$

Alert Propagation Latency measures the time required for an attack detection to propagate across the UAV fleet:

$$\Delta t_{alert} = t_{fleet_alert} - t_{detect} \quad (3.21)$$

Leader Election Latency evaluates the time required to restore coordination after cluster head failure:

$$\Delta t_{election} = t_{announce} - t_{last_heartbeat} \quad (3.22)$$

Detection Continuity Rate (DCR) measures the robustness of detection during leadership transitions:

$$DCR = \frac{E_{detected}}{E_{attack}} \quad (3.23)$$

False Positive Rate (FPR) and **False Negative Rate (FNR)** quantify operational reliability by measuring false alarms and missed detections.

These metrics are widely used in intrusion detection systems and distributed security architectures to evaluate both detection quality and system reliability [45].

3.12 Conclusion

This chapter described the complete methodology used to build the proposed UAV intrusion detection system. It covered the adopted E-DIDS-inspired architecture, the internal system modules, and the organization of UAV fleets into clusters for efficient coordination and monitoring.

It also detailed the data collection and preprocessing pipeline, as well as the design of the Kolmogorov-Arnold Network (KAN) classifier used for intrusion detection. Furthermore, the chapter introduced the different intrusion detection subsystems and the overall system workflow, ensuring a clear understanding of how detection is performed at both subsystem and global levels.

Finally, the evaluation metrics were presented to define how the performance of the proposed system is measured in terms of both classification quality and operational efficiency. This methodology lays the foundation for the implementation and experimental results discussed in the following chapter.

Chapter 4

Chapter 4: System Implementation and Performance Evaluation

4.1 Introduction

This chapter presents the experimental evaluation of the proposed Exhaustive Distributed Intrusion Detection System (E-DIDS) across multiple UAV communication layers. The evaluation is conducted on three distinct intrusion detection subsystems targeting network-level traffic, navigation and telemetry exchanges, and embedded UAV communication protocols.

The results are analyzed in terms of classification performance, convergence behavior, and robustness across different attack scenarios. Furthermore, a comparative study is provided to highlight the effectiveness of the Kolmogorov--Arnold Network (KAN)-based architecture in handling heterogeneous UAV security threats.

In addition to quantitative results, a detailed discussion is provided to interpret model behavior, analyze limitations in specific attack contexts, and assess the suitability of the proposed approach for real-world UAV deployments.

4.2 Implementation

This section presents the implementation details of the proposed intrusion detection system, focusing on the core algorithms that drive the system's behavior. The implementation translates the theoretical framework introduced in the previous chapters into executable procedures that operate across the different components of the system.

In particular, the system is implemented as a set of coordinated algorithmic modules corresponding to the intrusion detection pipeline. These modules include data preprocessing, subsystem-based feature handling, model training, inference, and decision-making processes within the intrusion detection framework.

To ensure clarity and reproducibility, each major component of the system is described using structured pseudo-code. These algorithms illustrate the step-by-step logic followed by the system during both training and runtime phases, including how data flows through the intrusion detection subsystems and how final classification decisions are produced.

The following subsections present the main algorithms used in this work, starting from data preprocessing and ending with the final intrusion detection and decision-making process.

4.2.1 System Modules

The E-DIDS UAV system is composed of multiple cooperative modules that operate in a distributed architecture. Each module is responsible for a specific function ranging from sensing and detection to decision-making and coordination.

Sensor Module

The sensor module is responsible for collecting raw UAV data and transforming it into structured feature vectors for IDS processing.

Algorithm 1: Sensor Module Operation

Input: Raw UAV data stream, sampling interval

Output: Feature vectors

- 1 Initialize sensor and load dataset stream;
 - 2 **while** *system is running* **do**
 - 3 Read next raw data sample;
 - 4 Apply preprocessing and feature extraction;
 - 5 Push feature vector to IDS engine queue;
 - 6 Wait for sampling interval;
-

IDS Engine and Communication Module

The IDS engine performs inference on extracted features, while the communication module handles message exchange between UAV nodes.

Algorithm 2: IDS Engine and Communication Module**Input:** Feature vectors, trained model**Output:** IDS reports and network messages

```

1 while feature queue is not empty do
2   Retrieve feature vector;
3   Perform classification using trained model;
4   Compute confidence score;
5   if attack detected then
6     Create IDS report;
7     Send report to Cluster Head via communication module;

```

Central Control Module (Cluster Head Only)

This module is active only on the Cluster Head and performs multi-IDS fusion and global decision-making.

Algorithm 3: Central Control Module (Fusion and Decision)**Input:** IDS reports from UAV swarm**Output:** Fleet-wide alert or normal state

```

1 while reports are received do
2   Aggregate network, MAVLink, and UAVCAN reports;
3   Compute mean confidence per IDS group;
4   Determine number of active IDS groups;
5   if 3 groups active then
6     Compute fused score as average of all groups;
7     Set threshold = fusion threshold (3 IDS);
8   else
9     if 2 groups active then
10      Compute fused score as average of two groups;
11      Set threshold = fusion threshold (2 IDS);
12    else
13      Use single IDS score;
14      Set threshold = single IDS threshold;
15   if fused score  $\geq$  threshold then
16     Generate threat object;
17     Log incident in reporting module;
18     Broadcast FLEET_ALERT to all nodes;

```

Management Module

The management module handles node activity monitoring and policy dissemination.

Algorithm 4: Management Module Operation

Input: Node activity updates

Output: Policy updates

```
1 while system is active do
2   Collect node activity statistics;
3   Update node status table;
4   if policy update required then
5     Generate new policy;
6     Broadcast policy update to UAV agents;
```

Reporting Module

The reporting module aggregates system logs and computes performance metrics.

Algorithm 5: Reporting Module Operation

Input: System events and IDS results

Output: Performance summaries

```
1 while system is running do
2   Store events in circular buffer;
3   Update counters (alerts, errors, accuracy);
4   Log incidents if detected;
5 Generate summary report on demand;
```

4.2.2 Cluster Head Election Protocol

The Cluster Head (CH) election is based on a composite metric that combines both residual energy and behavioral trust. Each UAV independently computes the same deterministic decision function using locally available information, ensuring consistency across the swarm without negotiation.

Algorithm 6: Energy and Trust-Based Cluster Head Election**Input:** Node table $T = \{(b_i, t_i)\}$ where b_i is battery percentage and t_i is trust score**Output:** Selected Cluster Head node ID

```

1 foreach node  $i \in T$  do
2   | Normalize energy:  $e_i \leftarrow \frac{b_i}{100}$ ;
3   | Compute composite score:  $S_i \leftarrow e_i \times t_i$ ;
4 Select winner:  $CH \leftarrow \arg \max_i S_i$ ;
5 if multiple nodes have same score then
6   | Select node with smallest ID;
7 return  $CH$ ;

```

4.2.3 Preprocessing

This stage defines the unified preprocessing pipeline applied to all UAV intrusion detection datasets. It ensures data cleaning, normalization, feature selection, and class balancing before model training.

Algorithm 7: Unified Preprocessing Pipeline for UAV IDS Datasets**Input:** Raw dataset D , label column y , number of selected features k **Output:** Processed training, validation, and test sets**1 Step 1: Load and Clean Data**

- 2 Load dataset D
- 3 Remove ID-based columns
- 4 Replace ∞ and $-\infty$ with NaN
- 5 Drop missing values

6 Step 2: Encoding

- 7 Encode categorical features using Label Encoding
- 8 Encode target labels y

9 Step 3: Train/Validation/Test Split

- 10 Perform stratified split of D :
- 11 70% Train, 10% Validation, 20% Test

12 Step 4: Feature Scaling

- 13 Fit StandardScaler on training set only
- 14 Transform validation and test sets using fitted scaler

15 Step 5: Feature Selection (RFE)

- 16 Apply RFE with Decision Tree estimator on training set only
- 17 Select top k features $\rightarrow$ feature mask M
- 18 Apply M to validation and test sets

19 Step 6: Data Balancing (SMOTE)

- 20 Apply SMOTE only on training set: Generate synthetic minority samples
- 21 Balance class distribution to majority class size

22 Step 7: Output

- 23 Return: X_{train}, y_{train} (SMOTE-balanced)
- 24 X_{val}, y_{val}
- 25 X_{test}, y_{test}
- 26 Saved scaler, encoder, and feature mask artifacts

SMOTE

To address class imbalance in the training dataset, the Synthetic Minority Over-sampling Technique (SMOTE) is applied. It generates synthetic samples for minority classes by interpolating between existing instances and their nearest neighbors in feature space.

Algorithm 8: SMOTE oversampling procedure**Input:** Feature matrix $X \in \mathbb{R}^{N \times d}$, label vector $y \in \mathbb{R}^N$, number of neighbors k **Output:** Balanced dataset (X_{bal}, y_{bal})

```

1 classes, counts  $\leftarrow$  Unique( $y$ )
2  $N_{maj} \leftarrow \max(\text{counts})$ 
3  $X_{list} \leftarrow [X]$ ,  $y_{list} \leftarrow [y]$ 
4 foreach  $c \in \text{classes}$  do
5   if  $\text{count}(c) = N_{maj}$  then
6     continue
7    $needed \leftarrow N_{maj} - \text{count}(c)$ 
8    $X_c \leftarrow X[y = c]$ 
9    $synthetic \leftarrow \emptyset$ 
10  for  $i \leftarrow 1$  to  $needed$  do
11     $base \leftarrow X_c[\text{RandomInt}(0, |X_c|)]$ 
12     $dists \leftarrow \text{EuclideanDistance}(base, X_c)$ 
13     $dists[base] \leftarrow +\infty$ 
14     $NN_k \leftarrow \text{argsort}(dists)[1 : k]$ 
15     $neighbor \leftarrow NN_k[\text{RandomInt}(0, k)]$ 
16     $\lambda \leftarrow \text{Uniform}(0, 1)$ 
17     $synthetic_i \leftarrow base + \lambda \cdot (neighbor - base)$ 
18    Append( $synthetic$ ,  $synthetic_i$ )
19  Append( $X_{list}$ ,  $synthetic$ )
20  Append( $y_{list}$ , FullVector( $needed$ ,  $c$ ))
21  $X_{bal} \leftarrow \text{VStack}(X_{list})$ 
22  $y_{bal} \leftarrow \text{Concat}(y_{list})$ 
23 return  $\text{Shuffle}(X_{bal}, y_{bal})$ 

```

4.2.4 Intrusion Detection Systems

This section presents the proposed Intrusion Detection System (IDS) framework used in this work. The system is built around a Kolmogorov-Arnold Network (KAN) classifier and is applied across three UAV communication environments: Network-based traffic, MAVLink protocol traffic, and UAVCAN protocol traffic. Each IDS follows a unified training pipeline with dataset-specific configurations.

Kolmogorov-Arnold Network (KAN)

The core learning component of the proposed IDS is the Kolmogorov-Arnold Network (KAN), a spline-based neural architecture designed to model complex nonlinear feature interactions with improved interpretability and stability.

Algorithm 9: KAN-Based Intrusion Classification Model

Input: Input feature vector X , Trained KAN model

Output: Predicted label y

- 1 Normalize input features X ;
 - 2 Initialize hidden representation $h \leftarrow X$;
 - 3 **foreach** *KAN layer* l **do**
 - 4 Compute B-spline basis expansion $B_l(X)$;
 - 5 Compute nonlinear transformation::
 - 6 $\phi_l(X) = B_l(X) \cdot W^{(spline)} + \sigma(X) \cdot W^{(base)}$;
 - 7 Aggregate features::
 - 8 $h \leftarrow \sum_i \phi_l(x_i)$;
 - 9 Apply Layer Normalization for stability;
 - 10 Compute output logits Z ;
 - 11 Apply Softmax: $P = \text{softmax}(Z)$;
 - 12 Predict class: $y = \arg \max(P)$;
 - 13 **return** predicted label y ;
-

The KAN model improves classical neural approaches by replacing fixed activation functions with learnable spline-based transformations, enabling better adaptation to heterogeneous UAV communication patterns.

Network IDS Training Pipeline

The Network IDS is designed to classify general network traffic into multiple attack categories using the improved KAN classifier.

Algorithm 10: Network IDS Training Procedure**Input:** Dataset $D_{network}$, Feature selection via RFE, SMOTE-balanced training set**Output:** Trained Network IDS model

- 1 Load and preprocess dataset;
- 2 Apply Recursive Feature Elimination (RFE);
- 3 Split data into training, validation, and test sets;
- 4 Apply SMOTE only on training set;
- 5 Initialize KAN classifier with architecture [64, 32];
- 6 Adapt spline grid using training data percentiles;
- 7 **foreach** *training epoch* **do**
 - 8 **foreach** *mini-batch* **do**
 - 9 Forward pass through KAN;
 - 10 Compute label-smoothed cross-entropy loss;
 - 11 Backpropagate gradients;
 - 12 Clip gradients to stabilize training;
 - 13 Update parameters using AdamW optimizer;
- 14 Evaluate model on validation and test sets;
- 15 Compute Accuracy, Macro-F1, Precision, Recall, ROC-AUC;
- 16 **return** trained Network IDS model;

MAVLink IDS Training Pipeline

The MAVLink IDS focuses on detecting anomalies in MAVLink-based UAV command and telemetry communication. The same KAN architecture is used, with dataset-specific feature selection.

Algorithm 11: MAVLink IDS Training Procedure**Input:** Dataset D_{maavlink} , Selected features via RFE ($n = 10$)**Output:** MAVLink IDS model

- 1 Preprocess MAVLink dataset;
- 2 Perform stratified split (70% train, 10% validation, 20% test);
- 3 Apply SMOTE to training data only;
- 4 Initialize KAN classifier [64, 32];
- 5 Adapt grid boundaries from training distribution;
- 6 **foreach** *epoch* **do**
 - 7 **foreach** *batch* **do**
 - 8 Forward propagation through KAN;
 - 9 Compute classification loss;
 - 10 Backpropagate and update parameters;
- 11 Evaluate model using classification report and ROC-AUC;
- 12 **return** MAVLink IDS model;

UAVCAN IDS Training Pipeline

The UAVCAN IDS is designed for large-scale UAV swarm communication monitoring using the UAVCAN protocol. Due to dataset size, a larger batch size is used for efficiency.

Algorithm 12: UAVCAN IDS Training Procedure**Input:** Dataset D_{uavcan} , Feature set selected via RFE ($n = 8$)**Output:** UAVCAN IDS model

- 1 Preprocess UAVCAN dataset;
- 2 Perform stratified split (70% train, 10% validation, 20% test);
- 3 Apply SMOTE on training data only;
- 4 Initialize KAN classifier [64, 32];
- 5 Set batch size to 1024 for scalability;
- 6 **foreach** *epoch* **do**
 - 7 **foreach** *batch* **do**
 - 8 Forward pass through KAN;
 - 9 Compute loss function;
 - 10 Backpropagate gradients;
 - 11 Apply gradient clipping;
 - 12 Update model parameters;
- 13 Evaluate using Accuracy, Macro-F1, and ROC-AUC;
- 14 **return** UAVCAN IDS model;

4.3 Results Discussion

IDS Subsystem	Classes	Accuracy	Macro F1	ROC-AUC
Network IDS (UAV Traffic Layer)	Blackhole, Flooding, Sybil, Wormhole, Normal	0.9333	0.9350	0.9944
Navigation & Telemetry IDS (MAVLink)	GPS Jamming, GPS Spoofing, Normal	1.0000	1.0000	1.0000
Embedded Communication IDS (UAVCAN)	Flooding, Fuzzy, Normal	0.9648	0.9614	0.9909

Table 4.1: Overall performance comparison of the proposed E-DIDS architecture across UAV communication layers.

4.3.1 Network Intrusion Detection System Results

This subsection presents the experimental results obtained for the Network Intrusion Detection System (NIDS), which is responsible for detecting malicious behavior in UAV network traffic using a Kolmogorov-Arnold Network (KAN)-based classifier.

Training Convergence Analysis

The training process shows stable convergence across epochs. The training loss decreases steadily while validation accuracy improves consistently, reaching more than 93% in later epochs.

This indicates that the model generalizes well to unseen data and does not suffer from significant overfitting. The evolution of the F1-score confirms balanced performance between precision and recall during training.

Metric	Score
Accuracy	0.9333
Macro Precision	0.9383
Macro Recall	0.9347
Macro F1-score	0.9350
ROC-AUC	0.9944

Table 4.2: Overall performance of the Network IDS on the test set.

Class	Precision	Recall	F1-score	Support
Blackhole Attack	0.9534	0.8371	0.8915	5225
Flooding Attack	0.9567	0.9519	0.9543	3947
Normal Traffic	0.9841	0.9788	0.9814	5237
Sybil Attack	0.9684	0.9662	0.9673	4818
Wormhole Attack	0.8289	0.9395	0.8807	5220

Table 4.3: Per-class performance of the Network IDS.

Confusion Matrix Analysis The classification behavior of the Network IDS is illustrated in Fig. 4.1.

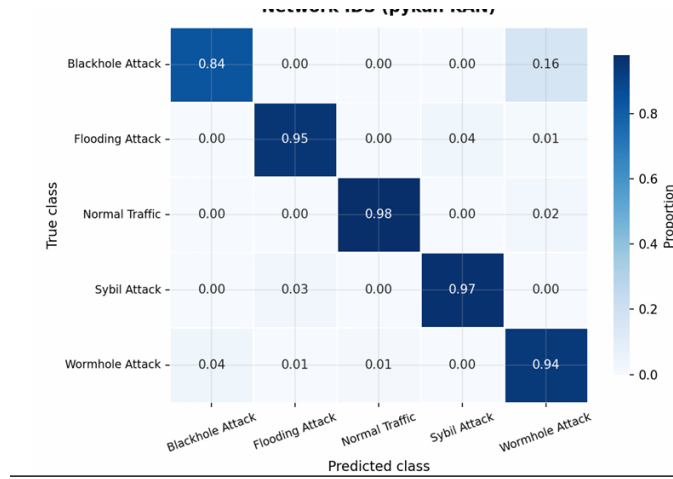


Figure 4.1: Confusion matrix for Network IDS (Blackhole, Flooding, Sybil, Wormhole, Normal).

Discussion

The obtained results demonstrate that the Network Intrusion Detection System achieves strong and stable performance across all attack categories. With an overall accuracy of 93.33% and a macro F1-score of 0.935, the model shows balanced detection capability across both normal and malicious traffic.

Best performance is observed for Flooding and Sybil attacks, where both precision and recall exceed 95%. This indicates that these attack patterns are more easily separable in the feature space learned after preprocessing and feature selection.

In contrast, the Blackhole attack class shows lower recall (0.8371), suggesting that some instances are misclassified as other traffic types. This may be due to behavioral similarity between Blackhole attacks and normal routing behavior in UAV networks.

Despite this, the ROC-AUC score of 0.9944 confirms that the model maintains excel-

lent discriminative capability across decision thresholds. Overall, these results validate the effectiveness of the KAN-based approach for UAV network intrusion detection.

4.3.2 Navigation and Telemetry Intrusion Detection System Results

This subsection presents the experimental evaluation of the Navigation and Telemetry Intrusion Detection System (MAVLink IDS), which is responsible for detecting GPS-related anomalies such as jamming and spoofing using MAVLink telemetry features and a Kolmogorov-Arnold Network (KAN)-based classifier.

Quantitative Performance

The MAVLink IDS achieves the following performance on the test set:

Metric	Value
Accuracy	1.0000
Macro Precision	1.0000
Macro Recall	1.0000
Macro F1-score	1.0000
ROC-AUC	1.0000
Mean Confidence	0.9481
Confidence Std	0.0065

Table 4.4: Overall performance of the MAVLink IDS.

Class	Precision	Recall	F1-score	Support
GPS Jamming	1.0000	1.0000	1.0000	292
GPS Spoofing	1.0000	1.0000	1.0000	100
Normal	1.0000	1.0000	1.0000	1623

Table 4.5: Per-class performance of MAVLink IDS.

Confusion Matrix Analysis The classification behavior of the MAVLink IDS is illustrated in Fig. 4.2.

Critical Analysis of Perfect Classification Performance

Although the MAVLink IDS achieves perfect classification performance across all evaluation metrics, this result should be interpreted with caution. Rather than indicating absolute model superiority, it reflects a high degree of separability in the underlying feature space.

The selected features, particularly HDOP, EPV, and noise-related telemetry signals, exhibit abrupt and highly structured variations under attack conditions. This creates near step-function separations between normal and malicious classes, effectively simplifying the classification task.

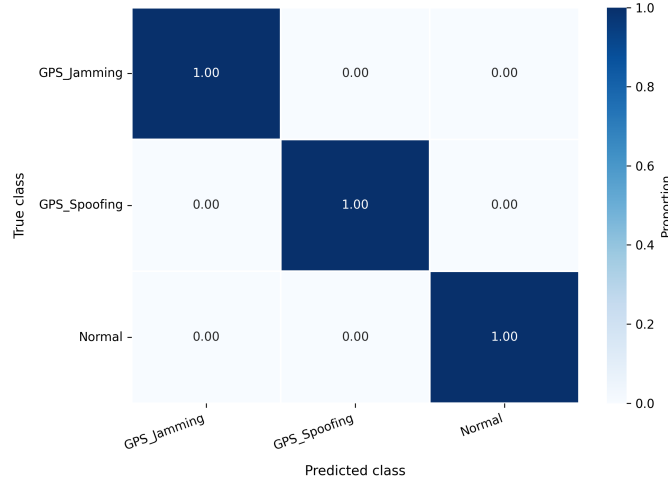


Figure 4.2: Confusion matrix for MAVLink IDS (GPS Jamming, GPS Spoofing, Normal).

From a modeling perspective, the Kolmogorov-Arnold Network (KAN) benefits from its spline-based edge functions, which can rapidly approximate sharp thresholds in feature distributions. As a result, the model converges quickly to near-deterministic decision boundaries without requiring complex hierarchical feature composition.

Additionally, the dataset construction introduces relatively deterministic attack patterns (e.g., GPS spoofing and jamming simulations with clear statistical deviation from normal telemetry). This reduces ambiguity in the learning problem and increases class separability.

Therefore, the observed 100% performance should not be interpreted as full real-world generalization capability, but rather as evidence that the MAVLink feature space in this dataset is highly structured and easily separable under supervised learning.

4.3.3 Embedded Communication Intrusion Detection System (UAVCAN)

This subsection presents the experimental results of the UAVCAN-based intrusion detection system operating at the embedded communication layer. The model is evaluated on byte-level message representations to detect *Flooding*, *Fuzzy*, and *Normal* communication behaviors.

Feature Space and Model Configuration Each sample is represented as raw UAVCAN byte-level features:

$$\mathbf{x} \in \mathbb{R}^8 \quad (4.1)$$

The final KAN architecture is defined as:

$$8 \rightarrow 32 \rightarrow 16 \rightarrow 3 \quad (4.2)$$

corresponding to the classes {Flooding, Fuzzy, Normal}.

Training Dynamics Table 4.6 summarizes the learning behavior over 10 epochs.

Epoch	Train Loss	Val Loss	Val F1
1	0.3745	0.1887	0.9450
2	0.3301	0.1737	0.9526
3	0.3229	0.1710	0.9516
4	0.3187	0.1657	0.9568
5	0.3158	0.1614	0.9591
6	0.3142	0.1592	0.9593
7	0.3112	0.1549	0.9615
8	0.3093	0.1559	0.9615
9	0.3081	0.1552	0.9620
10	0.3069	0.1530	0.9629

Table 4.6: UAVCAN IDS training progression

Metric	Value
Accuracy	0.9648
Macro Precision	0.9578
Macro Recall	0.9663
Macro F1-score	0.9614
ROC-AUC	0.9909
Mean Confidence	0.9187
Std Confidence	0.0557

Table 4.7: UAVCAN IDS test metrics

Test Performance

Confusion Matrix Analysis The classification behavior of the UAVCAN IDS is illustrated in Fig. 4.3.

Critical Discussion: Embedded-Layer Behavioral Learning Unlike navigation-layer datasets, UAVCAN operates at the embedded communication level where anomalies manifest as statistical perturbations in byte distributions rather than explicit semantic shifts.

The presence of both *Flooding* and *Fuzzy* attacks introduces two distinct learning challenges:

- **Flooding attacks:** characterized by repetitive high-frequency byte patterns that distort normal communication bandwidth usage.
- **Fuzzy attacks:** characterized by randomized payload corruption, introducing stochastic noise into the feature space.

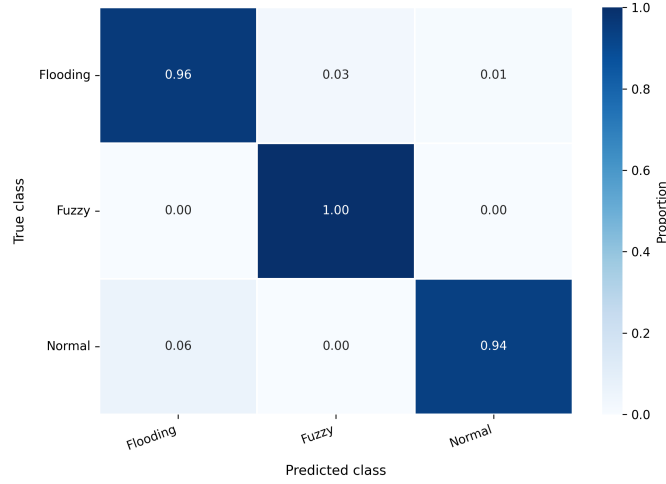


Figure 4.3: Confusion matrix for UAVCAN IDS (Flooding, Fuzzy, Normal).

This combination increases entropy in the input distribution and prevents trivial separability.

Model Interpretation (KAN Behavior) The Kolmogorov--Arnold Network adapts efficiently to this environment due to its spline-based edge functions, which are capable of modeling smooth transitions and irregular boundaries in byte-level feature distributions.

Unlike deterministic threshold models, the KAN does not rely on hard decision boundaries but instead learns continuous functional mappings over communication patterns. This is particularly effective for detecting fuzzy attacks, where anomalies are not defined by fixed thresholds but by probabilistic deviations.

Implication The achieved performance (Macro F1 = 0.9614) indicates strong generalization under noisy embedded conditions. However, the slightly reduced accuracy compared to higher-layer IDS modules reflects the intrinsic difficulty of detecting low-level protocol manipulation, where attack signatures are less deterministic and more distributionally ambiguous.

4.4 Limitations of the Work

Despite the experimental results achieved by the implemented E-DIDS framework, several limitations should be acknowledged.

First, part of the evaluation relies on curated and semi-synthetic datasets, which may not fully capture the complexity and unpredictability of real-world UAV flight environments. This is particularly relevant for scenarios involving adaptive adversaries and environmental noise variations.

Second, some attack types in the datasets exhibit clear structural patterns, which can lead to high separability and potentially overestimate model performance under controlled conditions.

Third, the current architecture evaluates each intrusion detection subsystem independently, without fully exploiting cross-layer correlations between network, navigation, and embedded communication layers.

Finally, the deployment aspect remains simulation-based, and real-time testing on physical UAV fleets has not yet been conducted, which limits validation under true operational constraints such as latency, packet loss, and hardware variability.

4.5 Conclusion

This chapter evaluated the performance of the proposed multi-layer intrusion detection framework across three UAV communication domains. The experimental results demonstrate that the KAN-based models achieve strong detection capabilities across all subsystems, with particularly high performance in navigation-level intrusion detection and robust generalization in embedded communication scenarios.

The analysis of training dynamics and confusion matrices confirms the model's ability to effectively learn discriminative representations for different attack types, while maintaining stable convergence across datasets.

Furthermore, the comparative results highlight the effectiveness of the proposed architecture in balancing detection accuracy and computational efficiency, making it suitable for resource-constrained UAV environments.

Conclusion

This thesis presented a distributed and multi-layer intrusion detection framework designed to enhance the security of UAV networks through intelligent, protocol-aware anomaly detection. By leveraging Kolmogorov--Arnold Networks (KANs), the proposed system was able to model complex relationships within network traffic, navigation telemetry, and embedded communication data.

The results demonstrate that a layered security approach is both effective and necessary for modern UAV systems, where threats can emerge at multiple levels of the communication stack. While the obtained performance is encouraging, this work also reveals important directions for future improvement.

Future work will focus on enhancing the robustness and realism of the proposed system by improving IDS model generalization and evaluating performance on real UAV flight datasets collected under operational conditions. In addition, expanding the detection capability to cover a broader range of attack classes will be essential to better reflect evolving adversarial behaviors.

Another important direction involves improving cluster-based organization mechanisms and optimizing the cluster head (CH) election protocol to strengthen distributed coordination within UAV swarms. Finally, a key long-term objective is the successful deployment of a fully dynamic IDS system directly on UAV networks, enabling real-time, adaptive intrusion detection in live aerial environments.

Overall, this work lays a solid foundation for intelligent UAV cybersecurity systems and opens the path toward more adaptive, scalable, and deployment-ready defense architectures.

References

- [1] A. A. Laghari, A. K. Jumani, R. A. Laghari, and H. Nawaz, ``Unmanned aerial vehicles: A review," *Cognitive Robotics*, vol. 3, pp. 1--11, 2023.
- [2] A. Chriki, H. Touati, H. Snoussi, and F. Kamoun, ``Fanet: Communication, mobility models and security issues," *Computer Networks*, vol. 163, p. 106877, 2019.
- [3] M. Mahbub, M. M. Saym, S. Jahan, A. K. Paul, A. Vahid, S. Hosseinalipour, B. Barua, H.-G. Yeh, R. M. Shubair, T. Taleb, A. Kaushik, M. H. Alsharif, M. S. Islam, R. R. Mahmud, and D. Niyato, ``UAV-assisted wireless communications in the 6G-and-beyond era: An extensive survey on characteristics, standardization and regulations, enabling technologies, challenges, and future directions," *Vehicular Communications*, vol. 56, p. 100977, 2025. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2214209625001044>
- [4] A. Hussain, S. Li, T. Hussain, X. Lin, F. Ali, and A. A. AlZubi, ``Computing challenges of UAV networks: A comprehensive survey," *Computers, Materials & Continua*, vol. 81, no. 2, pp. 1999--2051, 2024. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1546221824007677>
- [5] K.-Y. Tsao, T. Girdler, and V. G. Vassilakis, ``A survey of cyber security threats and solutions for uav communications and flying ad-hoc networks," *Ad Hoc Networks*, vol. 133, p. 102894, 2022.
- [6] M. S. Islam, A. S. Mahmoud, and T. R. Sheltami, ``Ai-enhanced intrusion detection for uav systems: A taxonomy and comparative review," *Drones*, vol. 9, no. 10, p. 682, 2025.
- [7] M. Hinduja, R. Kaushal, and S. K. Sandhu, ``A comprehensive survey on anomaly-based network intrusion detection systems," *Cybersecurity*, vol. 3, no. 1, pp. 1--32, 2020.
- [8] M. Ozkan-Okay, R. Samet, Ö. Aslan, and D. Gupta, ``A comprehensive systematic literature review on intrusion detection systems," *IEEE Access*, vol. 9, pp. 157 727--157 760, 2021.

-
- [9] Z. Liu *et al.*, "Kolmogorov-arnold networks," *arXiv preprint arXiv:2404.19756*, 2024.
[Online]. Available: <https://arxiv.org/abs/2404.19756>
 - [10] A. Al-Fuwaiers and A. K. Mishra, "Machine learning-based intrusion detection for drone IoT security," *Journal of Computer Information Systems*, vol. 14, no. 1, pp. 64--78, 2024.
 - [11] J. Whelan, A. Almeahmadi, and K. El-Khatib, "Artificial intelligence for intrusion detection systems in unmanned aerial vehicles," *Computers and Electrical Engineering*, vol. 99, p. 107784, 2022.
 - [12] O. Bouhamed, O. Bouachir, M. Aloqaily, and I. Al Ridhawi, "Lightweight IDS for UAV networks: A periodic deep reinforcement learning-based approach," in *Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management (IM)*. IEEE, 2021, pp. 736--741.
 - [13] M. Gamal, M. Elhamahmy, S. Taha, and H. Elmahdy, "Improving intrusion detection using LSTM-RNN to protect drones' networks," *Egyptian Informatics Journal*, vol. 27, p. 100501, 2024.
 - [14] A. H. Hamad, N. K. Hussein, and A. M. Abdulghani, "A deep learning paradigm for intrusion detection in unmanned aerial vehicle networks using extended LSTM," *International Journal of Intelligent Engineering and Systems*, vol. 18, no. 4, pp. 507--523, 2025.
 - [15] H. J. Hadi, Y. Cao, S. Li, F. Jamil, and M. N. M. Bhutta, "Real-time collaborative intrusion detection system in UAV networks using deep learning," *IEEE Internet of Things Journal*, vol. 11, no. 20, pp. 33 371--33 391, 2024.
 - [16] A. Wang, W. Wang, H. Zhou, and J. Zhang, "Network intrusion detection algorithm combined with group convolution network and snapshot ensemble," *Symmetry*, vol. 13, no. 10, p. 1814, 2021.
 - [17] R. T. Mehmood, G. Ahmed, and S. Siddiqui, "Simulating ML-based intrusion detection system for unmanned aerial vehicles (UAVs) using COOJA simulator," in *Proceedings of the International Conference on Open Source Systems and Technologies (ICOSST)*. IEEE, 2022, pp. 1--6.
 - [18] M. A. S. Rahman, M. T. Rahman, M. C. Kisacikoglu, and K. Akkaya, "Intrusion detection systems-enabled power electronics for unmanned aerial vehicles," in *Proceedings of the IEEE Workshop on the Internet of Safe Things (SafeThings)*. IEEE, 2021, pp. 1--6.

-
- [19] K. M. A. Alheeti, M. Alarfaj, D. Al-Dosary, and M. S. Al-Ani, "A hybrid security system for drones based on ICMetric technology," *PLOS ONE*, vol. 18, no. 3, p. e0282567, 2023.
 - [20] A. A. Khalil and M. A. Rahman, "Adaptive neuro-fuzzy inference system-based lightweight intrusion detection system for UAVs," in *Proceedings of the IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2023, pp. 1420--1425.
 - [21] L. M. da Silva, I. G. Ferrao, C. P. Dezan, D. Espes, and K. R. L. J. C. Branco, "Anomaly-based intrusion detection system for in-flight and network security in UAV swarm," in *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2023, pp. 384--393.
 - [22] R. Shrestha, A. Omidkar, S. A. Roudi, R. Abbas, and S. Kim, "Machine-Learning-enabled intrusion detection system for cellular connected UAV networks," *Electronics*, vol. 10, no. 13, p. 1549, 2021.
 - [23] J. Medhi, R. Liu, Q. Wang, and X. Chen, "A lightweight and efficient intrusion detection system (IDS) for unmanned aerial vehicles," *Neural Computing and Applications*, vol. 37, pp. 15 819--15 836, 2025.
 - [24] Y. Xie, W. He, H. Zhu, J. Liu, and J. Zheng, "A new unmanned aerial vehicle intrusion detection method based on belief rule base with evidential reasoning," *Heliyon*, vol. 8, no. 9, p. e10481, 2022.
 - [25] S. Ouiazzane, F. Barramou, and M. Addou, "Towards a multi-agent based network intrusion detection system for a fleet of drones," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 10, pp. 1--8, 2020.
 - [26] V. U. Ihekoronye, S. O. Ajakwe, D.-S. Kim, and J. M. Lee, "Hierarchical intrusion detection system for secured military drone network: A perspicacious approach," in *Proceedings of the IEEE Military Communications Conference (MILCOM)*. IEEE, 2022, pp. 761--766.
 - [27] F. Tlili, S. Ayed, and L. C. Fourati, "Exhaustive distributed intrusion detection system for UAVs attacks detection and security enforcement (E-DIDS)," *Computers & Security*, vol. 142, p. 103878, 2024.
 - [28] "Uavids2025 dataset," Kaggle Dataset, accessed: 2026-06-01. [Online]. Available: <https://www.kaggle.com/datasets/qinglizeng1997/uavids-2025>

-
- [29] "Uav mavlink intrusion dataset," IEEE DataPort, accessed: 2026-06-01. [Online]. Available: <https://ieee-dataport.org/open-access/uav-attack-dataset>
 - [30] "Uavcan intrusion dataset," IEEE DataPort, accessed: 2026-06-01. [Online]. Available: <https://ieee-dataport.org/documents/uavcan-intrusion-dataset>
 - [31] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," in *Machine Learning*. Springer, 2002, introduction of Recursive Feature Elimination (RFE).
 - [32] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers and Electrical Engineering*, vol. 40, no. 1, pp. 16--28, 2014.
 - [33] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321--357, 2002.
 - [34] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," *IEEE International Joint Conference on Neural Networks*, 2008.
 - [35] C. de Boor, *A Practical Guide to Splines*, revised edition ed. Springer, 2001.
 - [36] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," *arXiv preprint arXiv:1710.05941*, 2017. [Online]. Available: <https://arxiv.org/abs/1710.05941>
 - [37] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016. [Online]. Available: <https://arxiv.org/abs/1607.06450>
 - [38] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CVPR*, 2016.
 - [39] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *AISTATS*, 2010. [Online]. Available: <http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf>
 - [40] *MAVLink Developer Guide*, PX4 / ArduPilot Project, 2024. [Online]. Available: <https://mavlink.io/en/>
 - [41] *Cyphal (UAVCAN v1) Specification*, OpenCyphal Consortium, 2024. [Online]. Available: <https://opencyphal.org/>
 - [42] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

- [43] M. Ahmed, A. Mahmood, and J. Hu, ``A survey of network intrusion detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19--31, 2016.
- [44] T. Fawcett, ``An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861--874, 2006.
- [45] N. Shone, T. Ngoc, V. D. Phai, and Q. Shi, ``A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41--50, 2018.