UNIVERSITY OF SOUTHAMPTON

Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

An Investigation of Defence Protocols For the Mitigation of Grey Hole Attacks in Flying Ad Hoc Networks

by

Charles Edward James Hutchins
MEng
ORCiD: 0000-0002-8511-4423

A thesis for the degree of
Doctor of Philosophy

March 2026

University of Southampton

Abstract

Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

Doctor of Philosophy

An Investigation of Defence Protocols For the Mitigation of Grey Hole Attacks in Flying Ad Hoc Networks

by Charles Edward James Hutchins

Flying ad hoc networks (FANETs) are networks of unmanned aerial vehicles (UAVs) which are especially susceptible to subtle packet-dropping threats known as grey hole attacks (GHAs). GHAs vary in intensity and timing and can be easily confused with mobility-induced losses, which undermines detection systems and trust-based routing.

This thesis presents three principal contributions. First, a time-series FANET dataset is compiled and paired with an Early Time-Series Classification (ETSC) model that prioritises both detection accuracy and timeliness using node interaction histories and mobility features. Second, this thesis presents FANET-Rank, a game-theoretic evaluation framework that profiles defence protocols across multiple GHA threats and varying environmental conditions, producing robust and informative protocol rankings. Third, the Mobility-Conditioned Direct Trust (MCDT) mechanism establishes a trust-management method that conditions trust estimates on immediate mobility context as well as historical evidence to increase the reliability of trust scores. Simulations using NS-3 validate these contributions.

Contents

List of Figures

List of Tables

Declaration of Authorship

I declare that this thesis and the work presented in it is my own and has been generated by me as the result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. Parts of this work have been published as: Charles Hutchins, Leonardo Aniello, Enrico Gerding, and Basel Halak. A flying ad-hoc network dataset for early time series classification of grey hole attacks. *Scientific Data*, 12(1):1–22, 12 2025. ISSN 20524463. . URL https://www.nature.com/articles/s41597-025-05560-1 Charles Hutchins, Leonardo Aniello, Enrico Gerding, and Basel Halak. MANET-Rank: A Framework for Defence Protocols against Packet Dropping Attacks in MANETs. In *IEEE/IFIP Network Operations and Management Symposium*, 5 2024a

Signed:..... Date:.....

Acknowledgements

I would first like to express my sincere gratitude to my supervisors, Leonardo, Enrico, and Basel, for their guidance, patience, and encouragement throughout this work. Their insight and support have been invaluable in shaping both this research and my development as a researcher.

I would also like to thank my friends who have been on similar academic journeys, especially Spencer, Pughy, and Xander, for the many discussions, advice, and encouragement that helped me through the more challenging stages of this thesis.

To my other friends who have kept me grounded and sane outside the world of research, including Kip, Aarti, Alex, Tom, Rachel, and the rest of the Bath University crowd, thank you for the laughter, distractions, and constant support. I truly could not have done this without you.

To my parents, Peter and Honor, and my uncle Godfrey, for their endless love, support, and encouragement throughout every step of this journey.

Definitions and Abbreviations

Symbol	Description
α	The packet forwarding rate, biased towards more recent events.
β	The packet dropping rate, biased towards more recent events.
f	The total number of packets forwarded.
$\mathbb{d}$	The total number of packets dropped.
p	The probability of packet dropping for the probabilistic grey hole attack.
b	The probability that the selected time window will be a drop window in the time-based grey hole attack.
w	The length of the drop window.
q	The probability that a node will be assigned a drop status in the precursor-based grey hole attack.
s	A strategy which can be played by either player.
δ	The euclidean distance between nodes in m/s.
ψ	The direction from one node to another in m/s.
m	The current speed of the node in m/s.
X	A multi-feature time series (MTS) of time length τ.
$\bar{m}$	The highest node speed at which the watchdog mechanism can monitor sent packets. Used in the velocity-threshold protocol.
τ	The time length of an MTS.
$\mathbb{T}$	The maximum possible length of an MTS.
t	A point in discrete time, usually indicating the time at which a packet has been sent or received.
$\mathcal{Y}$	The set of class labels, defined depending on context.
y	A target data point within a dataset.
$\mathbf{Y}$	A collection of target data points within a dataset (can be class labels, or other data).
x	A feature data point within a dataset.

$\mathbf{X}$	A collection of features within a dataset (can be MTSs, or other data).
n	The number of data points within the dataset.
$\mathbf{D}$	A dataset.
V	The set of features.
v	A feature of the data.
$\mathcal{H}$	A base classifier.
$\mathcal{T}$	The set of classification points.
$\mathcal{F}$	An early classifier.
$\mathbb{R}$	The set of real numbers.
$\hat{.}$	A prediction from a classifier or model.
π	The performance of a base classifier.
C	The confidence of a base classifier prediction.
ϕ	A confidence threshold candidate.
Φ	The set of available confidence candidates.
κ	The trade-off between accuracy and Decision Latency.
L	A loss function.
Γ	A Normal Form game used in FANET-Rank.
N	The set of player types.
S	The set of player strategies.
s	A player strategy.
U	The set of utility functions, or payoffs, for all players.
a	The attacking player.
d	The defending player.
u	A utility function.
$\mathbb{M}$	A mapping from values to a normal form game.
r	The maximum regret.
Θ	The set of samples.
$\tilde{\Theta}$	A resampled set of samples.
B	The number of bootstrapped games generated.
γ	A discount factor applied to forwarding and dropping rates.
$\mathcal{M}$	A function which maps simulation data to a payoff matrix.
h	The time step corresponding to the oldest packet included in the lookback history.
λ	The predicted forwarding rate.
ω	The direct to indirect trust ratio.
σ	The forwarding status of a particular packet.
ρ	A direct trust measurement using the expectation of the beta distribution.

g	A trust threshold parameter.

Chapter 1

Introduction

Flying Ad hoc networks (FANETs) are collections of WiFi capable Unmanned Aerial Vehicles (UAVs) or "nodes" which autonomously build communication routes to provide services to large geographic areas. These UAV networks function as a cohesive entity, executing high-level tasks such as data collection, providing networked services, and dynamically coordinating missions. As a result, FANETs are widely used in applications such as precision agriculture, logistics, construction, and climate monitoring (Pasandideh et al., 2022; Benfriha et al., 2023).

One prominent application of FANETs is in search and rescue (SAR) operations, where they assist response teams in locating and aiding victims of natural disasters (Silvagni et al., 2016; Harris and Pooler, 2024). In these scenarios, rapid deployment and efficient coordination are critical, as timely intervention can mean the difference between life and death.

The UAVs within a FANET communicate by continuously updating their routing paths in response to changes in network topology. These routing paths, which may involve multiple intermediate nodes, enable the transmission of data across the network. The effectiveness of this communication is measured in terms of Quality of Service (QoS), which reflects the overall network performance and includes metrics such as packet delivery rate and end-to-end packet delay.

In the following sections, we will highlight the challenges these networks face in terms of cybersecurity in Section 1.1 and discuss their solutions, which form the main contributions of this thesis, in Section 1.2. To conclude the introduction, we will present the publications where this work has been disseminated in Section 1.3 and provide an outline of the overall thesis structure in Section 1.4.

1.1 Challenges

Most routing protocols in FANETs are designed under the assumption that all participating nodes behave honestly. However, this assumption can be exploited by attackers who compromise one or more nodes within the network (Chriki et al., 2019; Khanna and Sachdeva, 2019b). In particular, grey hole attacks (GHAs) pose a significant threat to QoS. In such attacks, malicious nodes participate in route establishment to appear legitimate but later drop data packets selectively or at random, thereby disrupting packet flow and degrading QoS. This makes GHAs very hard to reliably identify, as they can resemble benign packet dropping behaviour from benign nodes (Singh et al., 2021).

Predictably, the research community has proposed various countermeasures to detect and mitigate GHAs. Some of these works propose new defence protocols which use trust-based mechanisms (Cho et al., 2011) to evaluate a node's behaviour over time to determine the trustworthiness of the next hop. In addition, other defence protocols employ more sophisticated techniques such as Fuzzy Logic (Singh et al., 2020; Sadayan and Ramaiah, 2022), Game Theory (Vijayalakshmi et al., 2023; Subba et al., 2018) or Reinforcement Learning (RL) (Jinarajadasa et al., 2018; Ryu and Kim, 2023). Alternatively, some works specialise in intrusion detection systems (IDSs) (Alheeti et al., 2016; Hassan et al., 2024) which identify threats rather than proposing an entirely new protocol.

Despite these security efforts, GHAs still pose a significant challenge in FANET security due to their varied types, intensities, and their interaction with mobility. Specifically, changes in attack intensity, which refer to the aggressiveness of a node's packet-dropping behaviour, have only been investigated in a few studies (Bounouni et al., 2022; Tropea et al., 2024). Low-intensity attacks rely on stealth to evade detection, whereas high-intensity attacks cause significant disruption but are easier to identify and mitigate. Additionally, GHAs can also be classified by their type. For instance, probabilistic GHAs drop packets based on a predefined probability, time-based GHAs discard packets within specific time windows, and precursor-based GHAs drop packets depending on a node's identity. These types play a critical role in determining the effectiveness of FANET defences, yet their impact has not been systematically studied in detection, mitigation or evaluation.

Compounding this challenge is the high mobility of FANETs themselves: links can form and break within seconds, meaning that the same defence strategy may behave very differently depending on the movement patterns of neighbouring nodes. A defence that seems effective under static or low-mobility conditions may collapse when nodes move at higher speeds or cluster unevenly. Thus, both the characteristics of GHAs and the dynamics of node mobility must be considered together. Furthermore, all of the contributions in this thesis address the combined challenges of GHA types, GHA

intensities, and node mobility. Within this context, three further issues arise. These are introduced in Sections 1.1.1, 1.1.2 and 1.1.3 and are expanded upon in Chapters 4, 5 and 6.

Firstly, in Section 1.1.1, we examine the challenge of designing IDSs that can detect GHAs at the earliest possible stage. Because different GHA types and intensities require different defence responses, accurate detection is essential. Moreover, the demands of search and rescue operations require that detection be both rapid and reliable. However, many existing IDSs rely on aggregated or averaged sections of FANET log data, which delays detection because sufficient data must be collected before a result can be produced. Moreover, averaging can obscure local anomalies, making abnormal behaviour appear normal.

Secondly, in Section 1.1.2, this thesis addresses the challenge of selecting appropriate defence responses once an attack has been identified. A defence protocol must remain effective across the full range of GHA types and intensities, rather than being tailored to a single fixed scenario. Achieving this often requires accepting small reductions in QoS under one attack type in order to prevent severe degradation under another. For example, a protocol optimised exclusively for probabilistic grey hole attacks may achieve strong QoS in that scenario but perform poorly when facing time-based grey hole attacks; a more balanced protocol may sacrifice a little QoS against probabilistic attacks but remain robust against both. Yet, the analysis in this thesis shows that common evaluation practices often ignore this trade-off, instead favouring protocols that perform best under one threat condition.

Finally, in Section 1.1.3, this thesis addresses the limitations of current trust management approaches. In FANETs, trust management is used to evaluate and maintain the credibility of neighbouring nodes by monitoring their behaviour. However, existing schemes face significant challenges: trust values are often derived from incomplete or delayed information, and the methods for fusing observations over time can produce inaccurate or outdated assessments. These limitations risk misclassifying nodes and ultimately reduce the QoS of the network.

These three challenges — early detection, comprehensive evaluation, and accurate trust mechanisms — form the foundation of the investigations presented in this thesis.

1.1.1 Fast Intrusion Detection

In IDS systems, no existing work has optimised the accuracy of GHA identification as well as the speed at which GHAs are recognised. This means that previously proposed FANET IDS systems have limited real-world applicability, as they require observing large fixed time windows of FANET interactions before reaching a decision. In addition, progress in developing high speed GHA detection models is significantly hindered by

the lack of suitable datasets. Existing cybersecurity datasets are ill-suited for FANETs for several reasons. Firstly, popular cybersecurity datasets are often generated using environments which fail to reflect the unique topology and dynamics of FANETs (Stolfo Salvatore and Chan, 1999; Tavallaee et al., 2009; Sharafaldin et al., 2018). For example, many datasets are derived from Local Area Networks (LANs), where nodes are fixed in place, connected through stable wired or wireless links, and rarely experience changes in network structure. In contrast, FANET nodes are highly mobile, moving at significant speeds and frequently disconnecting and reconnecting as communication links form and break in real time. This fundamental difference means that datasets designed around LAN behaviour cannot capture the mobility-driven challenges of FANETs. This brings us to our first research question:

RQ1 How can datasets used for FANET network security be improved to better reflect the unique topology and dynamics of FANETs?

Secondly, these datasets typically assign a label to individual data points rather than to a sequential time series of interactions. As a result, models trained on such data cannot learn to refine their predictions over time, instead being forced to classify based on a single snapshot rather than an evolving pattern of behaviour. This brings us to our second research question:

RQ2 How can FANET datasets be structured to capture the sequential nature of node interactions, enabling models to learn evolving patterns of behaviour rather than relying on isolated snapshots?

Finally, these datasets seldom differentiate between various GHA types and intensities, making it challenging to develop models capable of distinguishing attack strategies with distinct behavioural patterns. This brings us to our third research question:

RQ3 To what extent do different GHA types and intensities exhibit unique behavioural patterns, and how feasible is it for an algorithm to reliably and rapidly distinguish between them in real-world FANET scenarios?

1.1.2 Comprehensive Evaluation of Defence Protocols

While the identification of GHAs is crucial, it does not inherently provide a suitable response mechanism. Determining the correct course of action following identification is far from trivial, as several challenges arise. For instance, should a node be permanently excluded from the network, or should a rehabilitation mechanism be implemented which allows previously identified malicious nodes to be reassessed and

potentially reintegrated? These types of decisions not only determine how a FANET mitigates threats but also how its overall performance, including packet delivery rate and end-to-end delay, is affected. Hence, there should be a focus on evaluating a protocol's impact on network performance as well as detection accuracy and speed.

In line with previous literature, a typical evaluation procedure involves simulating the proposed protocol alongside several baseline defence protocols under identical attack conditions. The protocols are tested in a simulated FANET environment with a fixed number of nodes, consisting of both defending and attacking nodes. The attack node ratio (the proportion of attacking nodes w.r.t the total number of nodes in the FANET), and the maximum node speed are treated as *environment parameters* and are varied to test the proposed protocol's resilience. Each simulation is executed multiple times using different seeds, which randomize the initial positions and velocities of nodes to ensure diverse testing scenarios. Once simulations are complete, the performance metrics are gathered and averaged per protocol. The expectation is that the proposed protocol will demonstrate superior performance when compared with the baseline defence protocols.

This evaluation procedure appears reasonable and robust; however, there are **four** overlooked issues which substantially limit the scope of the evaluation and can lead to incorrect conclusions regarding which protocol performs the best.

Unexplored Configuration Space. This work refers to *configuration parameters* as the changeable properties of a defence or attack protocol. For example, a model proposed by Singh et al. (2020) defines fuzzy membership functions based on Gaussian curves, which affects how their packet forwarding ratio influences their final decision. However, the authors do not explore the effects of implementing alternative membership functions, which could be practically implemented by modifying a configuration parameter. Configuration parameters may strongly impact protocol performance in some scenarios; therefore, it is important to evaluate the protocol under a range of parameters. *Defence configuration parameters* are typically chosen based on expert knowledge or empirical evidence (e.g. a trust threshold), whereas *attack configuration parameters* are commonly introduced as part of the threat model (e.g. % packet drop rates). Surprisingly, we find that previous work typically only tests a single or limited set of defence configuration parameters (Mukherjee et al., 2018; Sánchez-Casado et al., 2015; Wei et al., 2014; Bounouni et al., 2022; Singh et al., 2020; Sadayan and Ramaiah, 2022; Tropea et al., 2024).

Performance variability under different threats. Evaluating defence protocols and their configuration parameters against a broad spectrum of security threats offers significant advantages. However, it is unlikely that a defence protocol will perform well against every threat. According to previous evaluation procedures, a defence protocol can be deemed superior due to marginal gains under one attack, despite significantly poorer performance under another. This could lead to the selection of protocols that are

vulnerable in certain areas of the threat landscape. This brings us to our 4th research question:

RQ4 How can evaluation methodologies account for an expanded threat landscape and ensure that defence protocols, with their appropriate configuration parameters, remain effective regardless of how the adversaries act?

Performance Uncertainty. As previously noted, a common evaluation practice includes averaging the performance metrics across multiple seeded simulations. However, relying solely on average performance fails to account for the variability caused by randomized node placements and velocities. While the highest level of performance is a strong indicator of reliability, the lowest performance observed across runs is also of considerable interest, as it highlights the potential vulnerability of a protocol under adverse conditions. Together, these perspectives emphasise the need for evaluation methods that capture both performance and robustness. This brings us to our 5th research question:

RQ5 How can we take into account the variability of performance metrics under randomized node placements and velocities to ensure effective evaluation?

Unexplored Environment Space. In this thesis, we refer to specific *environments*, defined by two key parameters of the FANET: the attack node ratio (ANR) and the maximum node speed. The ANR denotes the proportion of attacking nodes relative to the total number of mobile nodes in the simulation, while the maximum node speed is the speed limit of the nodes within the FANET. There are a wide range of environment parameters which could be used to describe an environment, such as the choice of propagation loss model or mobility model. However, it has been noted that both ANR and maximum node speed are the most influential parameters during the evaluation of ad-hoc networks (Jari et al., 2021; Wei et al., 2014; Sánchez-Casado et al., 2015; Sadayan and Ramaiah, 2022; Singh et al., 2020; Bounouni et al., 2022; Rani et al., 2020; Ullah and Das, 2018), due to the fact that they make an exceptionally larger difference to the performance of the network compared with other environment parameters. Typical protocol evaluation involves varying one environmental factor while holding others constant (Singh et al., 2020; Bounouni et al., 2022; Jinarajadasa et al., 2018; Ullah and Das, 2018; Chandrasekar et al., 2024; Wei et al., 2014; Sadayan and Ramaiah, 2022; Jari et al., 2021). This approach leaves large regions of the environmental parameter space unexplored, which may obscure behaviours or weaknesses that only arise under specific combinations of conditions. This brings us to our 6th research question:

RQ6 To what extent does the maximum node speed and attack node ratio jointly affect the final evaluation result?

1.1.3 Reliable Trust Management

While early detection and comprehensive evaluation are critical, they do not in themselves guarantee effective defence. For this, an action must be taken by a UAV in order to retain the QoS of the network in the presence of GHAs. One such action is the exclusion of a suspicious UAV from the network. However, deciding whether that exclusion should occur depends not only on the UAV's maliciousness, but also on node mobility, since a node that is about to move out of range is inherently unreliable, regardless of its intent.

To support this decision, *trust management* (Benfriha et al., 2023) is employed, where trust values are assigned to UAVs based on their observed behaviour under the current interaction conditions. These values guide defence protocols by determining whether a UAV should be considered a reliable forwarding partner or excluded. Trust management can be divided into *direct trust* and *indirect trust*, where direct trust is formed from direct observations and indirect trust is based on 3rd hand information. This work considers direct trust only, since indirect trust is ultimately derived from the direct trust evaluations of other UAVs. It is therefore crucial that direct trust systems are both reliable and accurate.

However, current trust management approaches face two fundamental challenges. Firstly, direct trust values are typically derived from historic packet forwarding information alone, without considering the mobility of either node at the time of packet transmission. As a result, the trust value may not reflect the neighbour's true behaviour under the conditions in which the forwarding decision was made. Secondly, mobility information is typically gathered at trust update time, meaning that mobility information at time of packet forwarding is not considered.

Both challenges undermine the accuracy of trust values and risk misclassifying neighbours, either excluding benign nodes or permitting malicious ones. This can have severe consequences for the QoS of the network. This brings us to our final two research questions:

RQ7 How can direct trust mechanisms incorporate node mobility at the time of packet transmission so that trust values reflect the true interaction conditions between the current node and its neighbours?

RQ8 How can historic forwarding evidence be effectively fused with current mobility information at decision time to distinguish mobility-induced packet drops from malicious behaviour?

1.2 Research Contributions

To fulfil the 8 research questions, this thesis outlines 8 research contributions in three chapters of work. Each research question (RQ) directly aligns with a corresponding research contribution (RC) of the same number. Sections 1.2.1, 1.2.2 and 1.2.3 briefly describe these contributions.

1.2.1 A FANET Dataset for Fast GHA Detection

In Chapter 4, we introduce and motivate a novel dataset, FAN-GHETS24, designed for the fast classification of various GHAs. The dataset is derived from sequences of exchanged packet information and mobility-related data between two UAVs within the simulated FANET. These sequences undergo post-processing via two methods: firstly, an anonymization procedure that replaces IP addresses with standard string variables, allowing for offline model training and deployment on any UAV; and secondly, the application of feature engineering techniques to format the data for machine learning model integration. These contributions can be summarised as follows:

RC1 We introduce the FAN-GHETS24 dataset, a comprehensive dataset derived from FANET simulations, which includes both mobility data and packet information. This answers research question RQ1.

RC2 We structure FAN-GHETS24 to capture the sequential nature of node interactions by recording the time-series data of two-node interactions. This answers research question RQ2.

The dataset's utility is validated using a time series classification model, CEC-FANET, using early time-series classification (ETSC) methods. CEC-FANET successfully distinguishes between different GHA types and intensities with an accuracy of 70% while using an average of only 62% of the available sequence data. Compared to a fixed sequence length classification model, which uses 63% of the sequence, CEC-FANET shows a 5% improvement in accuracy. In addition, another fixed sequence length classification model, which uses 94% of the sequence, only improved upon the accuracy of CEC-FANET by 1%. This shows that ETSC methods can be used to both differentiate between GHA intensities and types whilst reducing the classification speed. This contribution can be summarised as follows:

RC3 Using FAN-GHETS24, we train CEC-FANET, a fast classification model that accurately differentiates between grey hole attack types and intensities while reducing classification time. This answers research question RQ3.

1.2.2 A Game-Theoretic Evaluation Framework for FANETs

In Chapter 5, we propose a new evaluation framework, FANET-Rank, which ranks defence protocols according to network performance while considering the strategies of the attacker. Specifically, empirical game theoretic analysis (EGTA) methods are used to build payoff tables consisting of network performance statistics. Then, decision theory is used to strategically assess the most effective defence protocol in the presence of multiple GHAs. Applying this technique revealed that a defence protocol, previously ranked highly by an older evaluation method, experienced a 12% loss in packet delivery rate when subjected to a different grey hole attack. Recognising this vulnerability, FANET-Rank appropriately lowered the protocol's ranking. In addition, evaluating a range of defence configuration parameters revealed a significant performance gap of 10% in packet delivery rate between two of them, highlighting the importance of testing multiple defence configuration parameters. This contribution is summarised as follows:

RC4 FANET-Rank uses empirical game-theoretic analysis and decision theory to rank defence protocols and their configuration parameters while accounting for attacker strategies. This answers research question RQ4.

The analysis of these payoff tables in RQ4 gives an indication of performance across the threat landscape. However, it does not take into account how node mobility in multiple simulation runs affects the values within the payoff table. In this work, we use bootstrapping to resample the original simulation values and produce performance distributions which take into account the uncertainty brought about by node mobility. As a result of bootstrapping, we observe differences in network QoS of up to 5%, highlighting that this method exposes meaningful differences in protocol performance. This contribution can be summarised as follows:

RC5 FANET-Rank resamples the original simulation runs using a statistical technique called bootstrapping which subsequently exposes the variability in performance metrics due to node mobility. This answers research question RQ5.

Broadening the evaluation further, we create an opportunity to test these protocols in a more diverse range of environments. Consequently, FANET-Rank shows that both the maximum node speed and the attack node ratio have a combined influence on network performance and should be evaluated together. This contribution can be summarised as follows:

RC6 FANET-Rank evaluates defence protocols across a two-dimensional environment parameter space, which confirms the influence of both

maximum node speed and attack node ratio on the final evaluation result. This answers research question RQ6.

1.2.3 A Mobility-conditioned Direct Trust Mechanism

Chapter 6 proposes a mobility-conditioned direct trust (MCDT) mechanism that estimates a neighbour's packet forwarding rate by combining maliciousness predictions with current mobility information. Maliciousness is first inferred from historical observations of mobility and packet forwarding behaviour. This historic evidence is then fused with current mobility conditions at decision time, producing a more accurate direct trust value.

MCDT is trained on interaction sequences from FAN-GHETS24 and evaluated using the FANET-Rank framework. Under this evaluation, MCDT achieves notable QoS gains over other direct trust mechanisms, with improvements of up to 12% in packet delivery rate and 75 ms in end-to-end delay. When compared with full trust-management systems that include indirect trust, MCDT delivers even larger gains in some environments, with up to 20% in packet delivery rate and 175 ms in end-to-end delay. Although, in this specific evaluation, it should be noted that MCDT did perform worse when considering all environments. This outcome is expected, as benchmarking a direct-trust-only mechanism, such as MCDT, against systems that also incorporate indirect trust, places MCDT at an inherent disadvantage. These contributions are summarised as follows:

RC7 We design a packet scoring model that evaluates historical packet transmissions in the context of node mobility at the time of sending, enabling a score to be generated which reflects the true behaviour of neighbouring nodes. This answers research question RQ7.

RC8 We develop a packet forwarding prediction model that fuses historical forwarding evidence with current mobility conditions at decision time. This answers research question RQ8.

1.3 Related Publications

Contributions RC1, RC2 and RC3 are described in the following journal article:

Hutchins C, Aniello L, Gerding E, Halak B. A flying ad-hoc network dataset for early time series classification of grey hole attacks. In: Nature, Scientific Data. 2025.

Contributions RC4, RC5 and RC6 are described in a conference paper and an extended journal article:

> *Hutchins C, Aniello L, Gerding E, Halak B. MANET-Rank: A Framework for Defence Protocols against Packet Dropping Attacks in MANETs. In: IEEE/IFIP Network Operations and Management Symposium. 2024.*
>
> *Hutchins C, Aniello L, Gerding E, Halak B. FANET-Rank: A Game Theoretic Evaluation Framework for FANETs Against Grey Hole Attacks. (Awaiting Venue)). 2025.*

1.4 Thesis Structure

Chapter 2 introduces the concept of a flying ad hoc network (FANET), outlining our specific application and the communication protocol employed.

In Chapter 3, we build our system model and simulation platform using the foundations established in Chapter 2. Following the creation of the system model, the threat model is established, which focuses on the grey hole attack. This includes a detailed description of the attack, potential scenarios, and the challenges associated with its detection. The chapter concludes with an overview of the simulation model utilized throughout the thesis, with minor variations specified in the relevant chapters. Additionally, we provide a detailed overview of our threat model. This system and threat model are used throughout chapters 4, 5 and 6.

In Chapter 4, we introduce FAN-GHETS24 and CEC-FANET to encourage the use of fast classification models to defend against GHAs. This chapter fulfils research contributions RC1, RC2 and RC3.

Chapter 5 presents the FANET-Rank evaluation framework which enhances traditional evaluation methods by incorporating a broader range of threats, environments, and parameters, thereby providing a more comprehensive assessment of defence protocols. This chapter fulfils research contributions RC4, RC5 and RC6.

Chapter 6 introduces MCDT, a mobility-conditioned direct trust mechanism that scores historic packet transmissions using mobility at send time and fuses this information with current mobility at decision time to produce accurate packet forwarding predictions. This chapter fulfils research contributions RC7 and RC8.

Chapters 4, 5, and 6 each begin with a brief introduction, including a motivation section that positions the work within the context of related studies. Finally, Chapter 7 concludes this thesis.

Chapter 2

Flying Ad-hoc Networks (FANETs)

Flying Ad-hoc Networks (FANETs) are collections of WiFi capable Unmanned Aerial Vehicles (UAVs) or nodes which autonomously build communication routes to provide network services to areas lacking fixed infrastructure. FANETs are increasingly being utilized across various application domains due to their versatility. Key applications include precision agriculture, goods delivery, the construction industry and climate monitoring (Pasandideh et al., 2022; Benfriha et al., 2023). We provide a brief overview of these application areas in Section 2.1 and conclude it by introducing our specific application: search and rescue (SAR).

As mentioned in the introduction, simulations are a necessary part of the research process when designing defence protocols for FANETs. And, to ensure that simulations remain as closely aligned with reality as possible, FANET specific elements are implemented in these simulations. To start, we describe UAV specific attributes used in SAR scenarios in Section 2.2; such as their maximum speed, size, weight and payload capacity. As we move through this Chapter, we begin to discuss more FANET specific attributes which may also encompass the UAV specifications mentioned previously. For example, Section 2.3 discusses the topological options of FANETs and their associated benefits with regards to their specific application areas.

As this work focuses on securing communication, we must give an explanation of FANET networking, which is provided in Section 2.4. Our work focuses on the security of FANETs, specifically concerning grey hole attacks, however, we also provide a broad overview of other FANET threats, each with a brief description and their classification in Section 2.8. While focusing on FANETs, we also reference other ad-hoc network paradigms, such as mobile ad-hoc networks (MANETs) and vehicular ad-hoc networks (VANETs), the justification for referencing these other paradigms is given in Section 2.7. Some proposed solutions to combat these threats are discussed in Section 2.9.

FIGURE 2.1: This figure shows some UAV footage taken inside a collapsed house in Noto in the aftermarth of the 1st January 2024 earthquake in Ishikawa Prefecture, Japan. From Yuki (2024).

2.1 Applications

In natural disaster scenarios, FANETs provide a flexible and safer alternative to the deployment of manned SAR teams. Moreover, manned SAR teams offer little benefit if small and hard to reach places need to be observed. A UAV's high manoeuvrability, speed, autonomy and small form factor are well suited to SAR scenarios (Mohammed Ahmed et al., 2021; Lyu et al., 2023; Pasandideh et al., 2022; Liu et al., 2023). This combination of factors make the use of UAVs an obvious and safe choice, as demonstrated recently in Brazil, where UAVs were used to locate flood victims (Harris and Pooler, 2024). Similarly, UAVs were also used to investigate the extent of the damage of the 1st January 2024 earthquake which struck Ishikawa Prefecture, Japan (Yuki, 2024). Figure 2.1 shows some drone footage of the rescue effort which highlights the small spaces that these UAVs need to be able to navigate to accomplish their mission. Moreover, UAVs have also shown their aptitude in specific SAR tasks. For example, they have proven their ability to assess building stability by using 3D point clouds (Levine and Spencer, 2022) or assess disaster scenes by using fast 3D modelling approaches (Verykokou et al., 2016). UAVs have also proven useful in the field of humanitarian logistics by transporting medical supplies to displaced people (Rejeb et al., 2021).

2.2 UAV Types

Due to the diverse SAR tasks described previously, UAVs of differing capabilities are required. Pasandideh et al. (2022) provide an in-depth review of UAV types and models, which are categorised by factors such as size, weight, communication range, endurance

UAV	Maximum Horizontal Speed	Maximum Takeoff Weight	Flight Time @ Maximum Take-off Weight
DJI Matrice 300 RTK (DJI, 2024)	23 m/s	9 kg	31 Minutes (Coptrz, 2024)
SkyFront Perimeter 8 (Skyfront, 2024)	15.8 m/s	23 kg	60 Minutes
Inspired Flight IF1200 (Inspired Flight, 2024)	25 m/s	25 kg	24 Minutes

TABLE 2.1: Table shows some examples of UAVs which could used for SAR operations

and applications. However, for conciseness, this work describes three UAV models of differing capabilities. The three identified UAV types are:

1. the DJI Matrice 300 RTK (DJI, 2024),
2. the SkyFront Perimeter 8 (Skyfront, 2024), and
3. the Inspired Flight IF1200 (Inspired Flight, 2024).

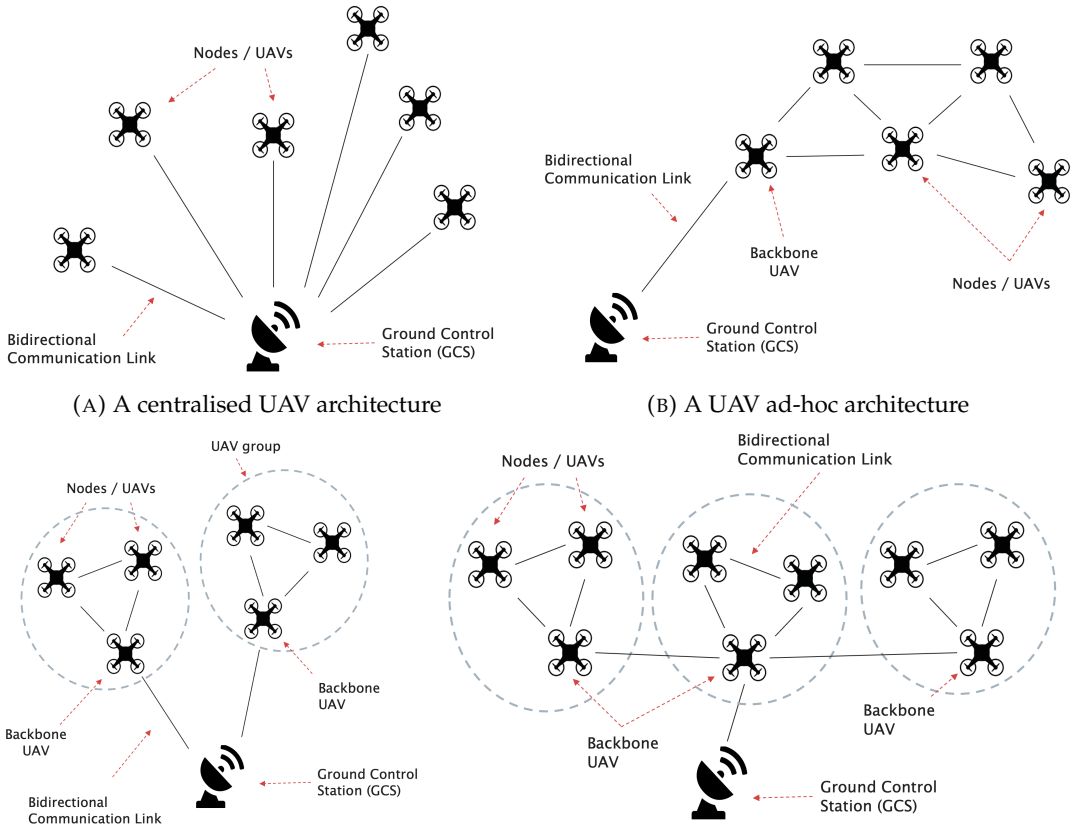
The SkyFront Perimeter 8 boasts a large carrying capacity of 23 kg with a comparatively long battery life of 60 mins compared with other UAV models. This makes the Skyfront Perimeter 8 suitable for humanitarian logistics tasks. On the other hand, the Inspired Flight IF1200 has a fast horizontal speed of 25 m/s, making it suitable for quickly assessing building stability. In contrast, the DJI Matrice 300 RTK boasts a compact design, enabling it to navigate through tight spaces with ease. Table 2.1 provides a summary of the UAVs.

2.3 Architectures

Li et al. (2013) presents several different architectures which exist in FANET research, of which, 4 are discussed here. Each architecture features a Ground Control Station (GCS) which issues commands to all UAVs in the FANET (Maxa et al., 2017).

In the centralized architecture, represented by Figure 2.2a, UAVs are connected through a single ground control station (GCS), which acts as the hub for all communication. This setup offers strong security since the GCS typically has greater computing power to run detection algorithms. However, such an architecture is not truly ad hoc and is unsuitable for long-distance operations.

In the UAV ad-hoc architecture, represented by Figure 2.2b, UAVs are connected directly with each other, forming a mesh or ad-hoc network where each UAV can communicate with any other UAV. These setups usually have a dedicated backbone



(A) A centralised UAV architecture

(B) A UAV ad-hoc architecture

(C) A multi-group UAV ad-hoc architecture

(D) A multi-layer UAV ad-hoc architecture

FIGURE 2.2: This figure shows 4 possible UAV architectures: centralised (Figure 2.2a), UAV ad-hoc (Figure 2.2b), multi-group (Figure 2.2c) and multi-layer (2.2d).

UAV which connects the ad-hoc UAVs to the GCS, this enables different protocols to be used for GCS-to-UAV communication and UAV-to-UAV communication. Alternatively, some ad-hoc networks do not use a backbone UAV at all (Khan et al., 2019), meaning that the GCS can communicate with any UAV in its range. This increases the number of potential routes through the FANET, but also fixes the network protocol for all communication links.

The multi-group UAV ad-hoc architecture, represented by Figure 2.2c, consists of groups of UAV ad-hoc networks, where each group has a backbone UAV which coordinates the cluster and communicates with the GCS.

The multi-layer UAV ad-hoc architecture, represented by Figure 2.2d, is an extension of the multi-group architecture, where all backbone UAVs can communicate directly with each other, forming an upper layer. The lower layer consists of all other UAVs within the network, which communicate within their group and with their designated backbone UAV.

2.4 Networking

Communication in a computer network (Tanenbaum et al., 2021) is implemented by exchanging payload data over a physical medium. To communicate this payload data clearly, protocol *headers* are layered on top of payload data which describes the ways in which the payload data is meant to be processed. The receiving device uses the headers in reverse order to reconstruct the original payload data for consumption. The Open Systems Interconnection (OSI) seven-layer model (International Organization for Standardization, 1994) provides a convenient conceptual framework for these layers and describes what happens at each layer. The list below gives a very short summary of the OSI model.

- **Application (Layer 7)** — end-user protocols and payloads (e.g. telemetry formats, video streams).
- **Presentation (Layer 6)** — optional transformations such as serialization, compression and encryption.
- **Session (Layer 5)** — session management which builds upon the transport layer.
- **Transport (Layer 4)** — end-to-end demultiplexing and optional reliability (UDP and TCP). Transport headers carry port numbers and checksums and are used to identify network flows.
- **Network (Layer 3)** — logical addressing and multi-hop forwarding (typically IPv4/IPv6). The network layer decides how packets are forwarded between hosts that are not directly adjacent.
- **Data link (Layer 2)** — framing for a single radio hop; contains MAC addresses, frame control bits and link-level reliability metadata (ACKs, retransmit counters).
- **Physical (Layer 1)** — This determines how bits propagate through the physical medium. It handles factors such as RF signalling, modulation, channel characteristics and bit error rates.

This section will expand on the network layer, as it is the most relevant for the content in this thesis. For a more comprehensive description of ad hoc routing at other layers, please refer to Perkins (2008).

Before describing FANET networking at the network layer of the OSI model, this section defines several terms which are used to express how one node relates to another during communication. For example, sending a packet to the next node in the route constitutes the *next hop* node. A *sender* and *receiver* imply a direct, one-hop connection where data is transmitted from the sender to the receiver without any intermediate

nodes. In contrast, an *origin* or *destination* node refers to the endpoints of a multi-hop connection, where data packets travel through one or more intermediate nodes before reaching their final destination. The *origin* node is the initial sender of the packets, while the *destination* node is the final receiver. A *neighbouring node* is any node that is within direct communication range of a given node and is capable of exchanging data without intermediate hops. Figure 2.3 provides a visual summary of these routing concepts, illustrating the roles of next hop, sender, receiver, origin, and destination nodes within a FANET.

The network layer determines how packets are routed from origin to destination. Moreover, in ad hoc networking, this route can be comprised of one or more hops, which indicate how many nodes a packet passes through in order to reach its destination. For successful routing to occur, both origin, destination and intermediary nodes need to be given a unique identifier so they are addressable in the network. These identifiers are called *Internet Protocol (IP) Addresses* and are used in *IP headers* to identify the source and destination node.

IP headers (and protocol headers in general) are a valuable source of information to security systems as they give some indication of network status. Table 2.2 gives a description of each field within an IP header. Some of these fields have been discussed previously (source and destination IP), although some fields provide more subtle information. Specifically, the *time-to-live (TTL)* field is initialized to a maximum value and decreases with each hop. Once this value reaches zero, the packet is discarded, preventing it from circulating indefinitely in the network. The protocol field is also significant, as it specifies how the packet should be processed. For example, a packet may be passed directly to the transport layer if it specifies the User Datagram Protocol (UDP) (Postel, 1980) or the Transmission Control Protocol (TCP) (Postel, 1981). Alternatively, the packet may be a *control packet*, in which case its contents are consumed by the network layer to support route maintenance.

In a typical local area network (LAN) a single router (the gateway) forwards traffic for every device and the Address Resolution Protocol (ARP) is used at the data link layer to learn the Media Access Control (MAC) address of a neighbour so frames can be sent. ARP is still used in ad hoc networks to determine the identity of next hop nodes, which suggests that malicious node identification and mitigation could be implemented at the data link layer. However, we focus on mitigation at the network layer as it provides access to route maintenance mechanisms, which can be leveraged to influence routing decisions. By contrast, the link layer can only block a specific MAC address from communicating, which ultimately results in delays at the network layer.

Field	Purpose / operational significance
Version + IHL	IP version and Internet Header Length (IHL). IHL indicates where the IP header ends and payload begins.
Type of Service (DSCP + ECN)	Differentiated Services Code Point (traffic class) and Explicit Congestion Notification bits; used for QoS and congestion signalling.
Total Length	Total length of the IP packet (header and payload); used to validate packet size and for fragmentation handling.
Identification	Identifier used when fragmenting packets; helps reassembly of fragmented payloads.
Flags and Fragment Offset	Fragmentation control (do-not-fragment, more-fragments) and offset for reassembly.
Time-To-Live (TTL)	Hop limit decremented at each forwarder to prevent routing loops; also useful as a crude indicator of path length.
Protocol	Encapsulated protocol identifier (e.g. 6=TCP, 17=UDP). Distinguishes application data from routing/control payloads.
Header Checksum	Integrity check for the IP header; validated by each hop.
Origin IP	Logical source address of the packet (end-to-end identifier used by routing decisions and addressing).
Destination IP	Logical destination address (remains constant across hops and indicates the ultimate recipient).
Options (if present)	Optional extensions (rare in typical data planes) that can carry route-recording, timestamping or other control semantics.

TABLE 2.2: This table shows the key IPv4 header fields and their operational significance.

2.5 Protocols

In contrast to traditional LANs, static routing tables cannot be used for traffic routing due to the constant mobility of FANET nodes. Therefore, we need to dynamically route data packets throughout the network, which is achieved by introducing routing messages to create and maintain links. Routes are updated and stored in a node's *routing table*, which specify the next hop IP for a particular destination IP. Entries in the routing table also include other information such as the validity and freshness of the route. The process of routing formation is controlled by a routing protocol, which can be classified into three categories:

Reactive These protocols create routes only when desired by the source node.

Proactive These protocols maintain fresh lists of destinations and their routes by periodically distributing routing tables throughout the network.

Field	Description
Version + IHL	IPv4 version and header length
Type of Service (DSCP+ECN)	Traffic class and congestion notification bits
Total Length	Length of IP packet (header + payload)
Identification	Fragment identification
Flags + Fragment Offset	Fragmentation control
Time-to-live (TTL)	Packet lifetime (hop limit)
Protocol	Encapsulated protocol (e.g. UDP=17, TCP=6)
Header Checksum	Header integrity check
Source IP	Source IPv4 address
Destination IP	Destination IPv4 address
Options (optional)	Optional extension fields

TABLE 2.3: IPv4 header fields and a brief description.

Hybrid These protocols combine the advantages of reactive and proactive protocols, utilizing both on-demand and periodic updates.

For each of these three categories, we will give an example ad-hoc protocol and give a brief description:

AODV Ad-hoc On-Demand Distance Vector (AODV) is a reactive routing protocol that establishes routes only when required by the source node. It uses route discovery and route maintenance mechanisms. When a source node needs to communicate with a destination, it broadcasts a Route Request (RREQ) message. Intermediate nodes propagate this message until it reaches the destination, which then responds with a Route Reply (RREP) message. If a link break occurs, a Route Error (RERR) message is sent to inform the affected nodes.

OLSR Optimized Link State Routing (OLSR) is a proactive routing protocol that maintains routes to all nodes in the network at all times. Each node periodically broadcasts Topology Control (TC) messages, which contain information about its neighbours and their link states. These messages are propagated throughout the network, allowing each node to maintain a complete view of the network topology.

ZRP Zone Routing Protocol (ZRP) is a hybrid routing protocol that combines the features of both reactive and proactive routing protocols. It divides the network into overlapping zones. Within each zone, proactive routing is used to maintain route information constantly. For inter-zone communication, ZRP uses reactive routing to establish routes on-demand. This combination allows ZRP to efficiently manage routing in large and dynamic networks by minimizing the overhead of route discovery while maintaining up-to-date routing information within zones.

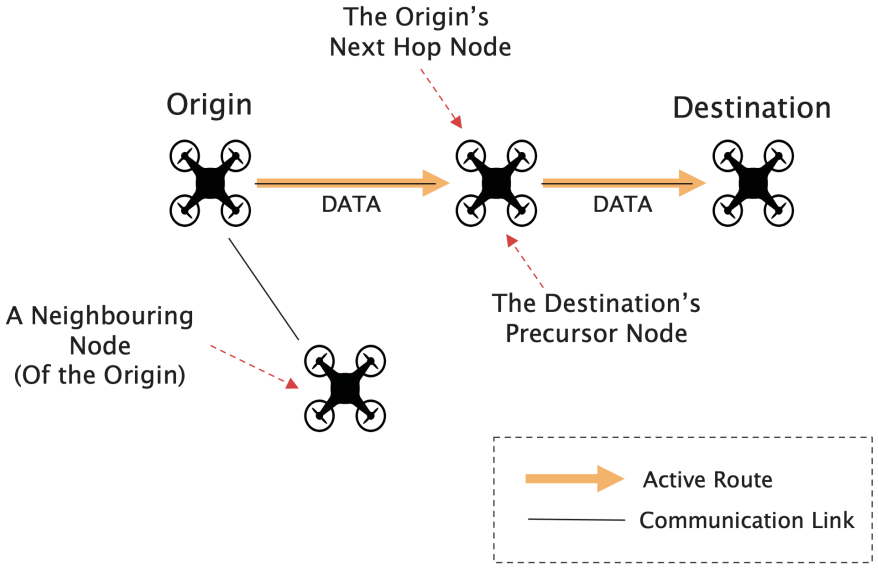


FIGURE 2.3: This Figure shows a simple 4 node FANET and the routing between the Origin and Destination node. Specific routing terms such as Next Hop, Precursor Node and Neighbouring Node are also indicated.

2.6 Performance Metrics

Performance metrics measure the performance of the FANET over the entire lifetime of the network either by aggregating or averaging measurable quantities in the network. To begin, we define the *data packet delivery rate* (DPDR), a performance metric that measures the efficiency of UDP data packet transmission by comparing the number of data packets sent by the originator node to the number of packets received by the destination node:

$$DPDR = \frac{\#Data\ Packets\ Received}{\#Data\ Packets\ Sent} \tag{2.1}$$

This metric is particularly useful during evaluation as it excludes control packets such as RREQs, RREPs, and RERRs. This performance metric deviates slightly from the traditional packet delivery rate (PDR) reported in the literature, which typically includes control messages. However, we find that focusing solely on data packets provides a clearer assessment of data transmission performance.

Contrary to the DPDR metric, we measure the *average end-to-end delay* (EED) and include control messages, as this offers a perspective on how defence strategies influence the delay in terms of route setup and maintenance. EED is calculated by measuring the time it takes for each packet to travel from the origin to destination node, divided by the total number of packets:

$$EED = \frac{\sum Packet\ Delay}{Total\ \#Packets} \tag{2.2}$$

It is worth emphasizing that these metrics are also utilized in studies which evaluate the performance of protocols for optimal path selection (Zhang et al., 2022) as well as protocols for security (Gurung and Chauhan, 2018). Given this context, we found it valuable to include a metric in our studies derived from the defence protocol's assessment of a node, specifically its ability to identify a malicious node. Therefore, we introduce the F1-Score (Ting, 2011), a metric well-suited for the evaluation of threat identification based on the defence protocol's outputs (Whelan et al., 2020). The F1 score balances precision and recall using their harmonic mean:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2.3}$$

where precision captures how many of the predicted malicious instances are actually malicious and recall captures how many correctly identified malicious instances there were compared with all malicious instances. The F1 score provides a single metric that considers both precision and recall, making it useful for evaluating a model's performance.

The performance metrics described so far (DPDR, EED and F1 score) will be used throughout this thesis, however, it is worth noting that three other performance metrics also exist, which will be referenced in later sections. Firstly, *Overhead* refers to the extra control packets needed for route discovery and maintenance. This metric is often given as a proportion of the total number of packets in the system. Secondly, *Jitter* is the variation in packet EED which is important for real-time applications since small fluctuations can disrupt the quality of service. Thirdly, *Energy* reflects the power used for transmitting and receiving packets. In battery-powered nodes, minimizing this performance metric is crucial to extend network lifetime and preserve connectivity.

2.7 Other Ad-hoc Network Paradigms

FANETs can also be classified as a sub category of Mobile Ad-hoc Networks (MANETs) (Benfriha et al., 2023; Chriki et al., 2019; Nayyar, 2018) or Vehicular Ad-hoc Networks (VANETs) (Bekmezci et al., 2013; Chriki et al., 2019; Nayyar, 2018). FANETs, VANETs and MANETs share common characteristics, such as the idea that the nodes of these network paradigms are all mobile to some degree, and, that all nodes of the network can act as sender, receiver and router. Due to these similarities, relevant works on MANETs and VANETs are also considered in order to provide a broader perspective on the ad hoc network landscape.

2.8 FANET Security

FANETs (and ad-hoc networks in general) are inherently vulnerable due to their dynamic topology, the implicit trust between nodes, decentralised nature and their reliance on wireless communication (Tsao et al., 2022). Moreover, cyber attacks can originate from two sources: insiders or outsiders. An insider leverages the inherent trust and access privileges of legitimate nodes, enabling them to more effectively disrupt network operations, intercept sensitive communications, and conduct attacks with reduced risk of detection. An outsider does not have these privileges, instead, they disrupt the normal workings of the network to gain unauthorised access or compromise the system's integrity without being a legitimate part of the network.

In this thesis, we focus on insider attackers, where an attacker has already penetrated the system by some means and now wishes to disrupt the network communications of the FANET. In Section 2.8.1 we state the general security requirements of a FANET and in Section 2.8.2, we outline some security threats which impact the network layer and subvert these security requirements. Definitions and descriptions in these sections are adapted from Tsao et al. (2022). Readers seeking additional detail should consult that work.

2.8.1 Requirements

Listed below are the security requirements for FANET network security.

- **Confidentiality** requires that sensitive information on or between nodes is disclosed only to authenticated and authorised parties on a need-to-know basis.
- **Integrity** ensures that data, commands, software, and logs are not modified without detection.
- **Availability** Ensures that network services and resources are accessible to the nodes when needed.
- **Privacy** ensuring that actions and communications within the network can be attributed to their source.

2.8.2 Threats

While this work focuses on the grey hole attack, we provide a brief description of other network layer related threats which pose a significant risk to FANETs:

- **Black Hole Attack**: In this attack, malicious nodes advertise having the shortest path to the destination, enticing other nodes to route their data through them. Once the data packets are received, the malicious node drops them instead of forwarding. This attack targets **availability** by creating a denial of service.
- **Grey Hole Attack**: Similar to the black hole attack, a grey hole attack involves a malicious node selectively dropping packets. However, grey hole attacks may not advertise shorter routes, but wait for legit routes to form before dropping packets. In addition, grey hole attacks may forward some packets to appear benign but drop others, making detection harder. Similar to black hole attacks, grey hole attacks target the **availability** of the FANET.
- **Worm Hole Attack**:In this attack, attackers create a tunnel or wormhole between two malicious nodes in the network, giving the illusion of a shorter path. This also allows the malicious nodes to steal packets or simply drop the packets completely. This affects the **confidentiality**, **integrity**, **availability** and **privacy** of the FANET.
- **Modification Attack**: This involves the manipulation or falsification of routing information. Attackers can disrupt the normal routing process, leading to incorrect routing decisions. Within this attack, several other threats are possible. For example, a *Sink Hole Attack* is a modification attack which sends route discovery messages quicker than neighbouring nodes, triggering the formation of a route with the malicious node included. Modification attacks affect the **Confidentiality**, **integrity** and **availability** of the FANET.

2.9 FANET Defence Mechanisms

A wide variety of defence mechanisms have been proposed to defend routing protocols like AODV in the face of adversarial behaviour. These mechanisms include cryptographic methods (Faraji-Biregani and Fotohi, 2021), trust-based systems (Singh and Verma, 2020; Benfriha et al., 2023; Tropea et al., 2024; Barka et al., 2018; Uma Rani et al., 2022; Chawhan et al., 2022), machine learning techniques (Singh and Verma, 2020; Kundu et al., 2024; Singh et al., 2020; Sbai and Elboukhari, 2022; Meddeb et al., 2023; Hanafi et al., 2023; Almomani et al., 2016; Ceviz et al., 2025b), reinforcement learning (Boyan and Littman·, 1993; Arafat and Moh, 2022; Rahul and Kaarthick, 2023; Ryu and Kim, 2023; Bouhamed et al., 2021), and game-theoretic models (Shila and Anjali, 2008; Lawrence and Latha, 2016; Subba et al., 2018; Vijayalakshmi et al., 2023). Cryptographic methods are an important part of network security, but are excluded from detailed discussion here, as this thesis focuses on behavioural, adaptive, and decision-making techniques rather than key management or message integrity.

In this Section, the discussion of various works in this field is organised by mechanism. Some works will therefore be mentioned in more than one section if they combine multiple techniques. For each defence mechanism, we briefly explain how the mechanism works and then go on to describe how this mechanism is used in specific pieces of work to defend against routing level attacks. This section distinguishes defence protocols, which act on the FANET to mitigate threats, from intrusion detection systems (IDSs), which detect attacks without intervening. This is an important distinction as the later is assessed by classification metrics such as F1 score, while protocols are assessed by network performance metrics such as DPDR and EED.

Trust management approaches are widely used for information gathering across mechanisms, so trust management is reviewed first in Section 2.9.1. Next, game theoretic approaches in Section 2.9.2 model the interaction between attacking and defending nodes. Reinforcement learning approaches, described in Section 2.9.3, learn from historical interactions within the FANET and choose actions accordingly. Other machine learning techniques, such as fuzzy logic and neural networks, are discussed in Section 2.9.4 and provide models that aggregate data and support decision making.

2.9.1 Trust Management

We start with the trust management mechanism, explored in a recent survey by Benfriha et al. (2023). Trust management is a process which establishes and updates the trust relationships between two UAVs from the perspective of a single UAV. A *trust value* quantifies this relationship by a real number between 0 and 1. A trust value of 0 signifies an untrustworthy relationship, whilst a trust value of 1 indicates full trust and cooperation. Figure 2.4 describes 5 stages of trust management as: information gathering, trust computation, trust aggregation, trust propagation and decision making.

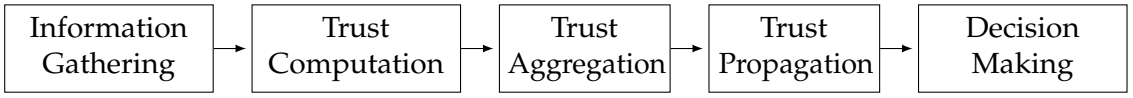


FIGURE 2.4: The 5 stages of Trust Management

The information gathering stage involves collecting data on node behaviour and past interactions. A popular way of gathering this data in the context of packet dropping attacks is to use a *watchdog mechanism*, which checks the forwarding behaviour of the next hop node by monitoring the node's incoming and outgoing packets. The watchdog mechanism plays a vital role throughout this thesis and is described in more detail in Section 2.9.1.1. In addition, other information is also gathered or measured at this stage, such as node energy, node mobility, transmission delay and third-party information in the form of *recommendations* from other UAVs.

Once the appropriate information has been gathered, it is then processed in the trust computation stage where *indirect* and *direct* trust values are computed. Direct trust is computed by combining all first hand information gathered in the information gathering stage. In contrast, computing indirect trust is harder because it requires both evaluating the trustworthiness of recommendations and aggregating them. More information in computing direct and indirect trust can be found in Sections 2.9.1.2 and 2.9.1.3 respectfully.

These trust values are combined in the trust aggregation stage to form an overall trust value. With this value, nodes can make informed decisions regarding the selection of nodes for routing and collaboration. This can be achieved by simply thresholding the overall trust value (Chawhan et al., 2022), or using fuzzy logic systems (Sadayan and Ramaiah, 2022).

Overall trust values are normally shared across the FANET in the trust propagation stage to provide other UAVs with more information regarding a particular node. The trust values are updated periodically, with historic information used as an information source in the next information gathering stage.

2.9.1.1 The Watchdog Mechanism

The information required by the trust management mechanism would be dependent on the type of security attack one wishes to subvert. Some examples of information used for trust management are packet drop rate, throughput or packet overhead Cho et al. (2011). Unsurprisingly, we see the use of packet drop rate in the majority of proposed works which defend against packet dropping attacks such as GHAs and BHAs (Tropea et al., 2024; Chawhan et al., 2022; Sedjelmaci et al., 2018). In fact, this packet drop information gathering system is given a specific name: the *watchdog* mechanism (Marti et al., 2000).

To introduce the watchdog mechanism, we give a brief example. Suppose that a defender node_j is sending a stream of packets to node_k. Using promiscuous mode, node_j can keep a record of the packets forwarded by all nodes in their sensing area, including the packets forwarded by node_k. As each node is uniquely identifiable and the next hop is always present in the routing table for a specific destination, these packet forwarding records can be reliably associated with a particular next hop node. Ergo, the packets forwarded by node_k can be compared with the packets sent originally by node_j to evaluate the number of packets forwarded, f_{jk}, and the number of packets dropped, d_{jk}. The tally of dropped and forwarded packets constitutes the watchdog mechanism. In order to determine whether a packet overheard from node_k is the one originally sent by node_j, node_j must store three fields: the Packet ID, the TTL at transmission, and the origin node's IP address. The Packet ID is unique only from the

origin's perspective, so different origins may reuse the same ID. For this reason, both the Packet ID and the origin IP must be matched when checking whether a packet has been forwarded. Node$_j$ should also store the TTL because its promiscuous sniffer may capture both the packet from node$_j$'s precursor and the packet forwarded by node$_k$. To confirm that node$_k$ forwarded node$_j$'s packet, the overheard packet from node$_k$ must match the Packet ID and origin IP, and its TTL must be exactly one less than the TTL of the packet transmitted by node$_j$.

While promiscuous mode is useful, it can reduce the energy efficiency of nodes because packets that would otherwise be discarded are now processed. Recent work integrates promiscuous monitoring with different architectures to improve threat detection. For example, HID-RS (Sedjelmaci et al., 2018) places the watchdog at the ground control station (GCS) rather than on the UAVs. The UAVs send their current routing information to the local GCS, which compares it with observed forwarding behaviour. Leveraging its greater range, the GCS can detect forwarded packets over a wider area. In turn, this increases the reliability of these trust values as nodes are less likely to disconnect from a local GCS compared with a neighbouring node. However, a key advantage of a UAV ad-hoc architecture is its ability to extend network coverage beyond the area directly visible to the GCS. Even though energy conservation is an issue in FANETs, one of the main advantages of a FANET is it's ability to extend the network coverage beyond what would be normally possible with simple ground antennas. Hence, for the rest of this thesis, we assume that UAVs have the ability to activate promiscuous mode and evaluate their neighbouring nodes.

2.9.1.2 Direct Trust Computation

While helpful, the forwarded and dropped packet counts from the watchdog mechanism do not produce a direct trust value. Some works employ the expectation of the beta distribution with discount factors (Ismail and Josang, 2002; Shabut et al., 2015) (which we refer to as ρ) to find the direct trust value of the next hop node.

$$T^D = \rho = \mathbb{E}[\text{Beta}(\alpha, \beta)] = \frac{\alpha}{\alpha + \beta}\,, \tag{2.4}$$

where

$$\begin{aligned} \alpha &\leftarrow \gamma_f \alpha + f_{jk}, \quad \gamma_f \in [0,1], \\ \beta &\leftarrow \gamma_{\mathrm{d}} \beta + \mathrm{d}_{jk}, \quad \gamma_{\mathrm{d}} \in [0,1]. \end{aligned} \tag{2.5}$$

The discount factors, γ_f and γ_{d} scale the forward and drop rates on every update of Equation 2.4, ensuring that the latest information has more impact. Moreover, the ratio of α and $\alpha + \beta$ means that the total number of packets is taken into consideration. For the first update, α and β take on an initial value of 1.

Other methods have also been used to calculate direct trust from f_{jk} and d_{jk}. Ryu and Kim (2023) update a trust value based on if the packet was successfully forwarded by the next hop as well as the distance to the next hop node and the time it takes for the packet to be received by the next hop node. They also integrate a decay factor in their direct trust update in a similar fashion to Equation 2.5. UNION, proposed by Barka et al. (2018), uses a reciprocal decay factor proportional to the total number of packets forwarded by the next hop node. This means that several packet forwards are required before any trust is earned.

Another method for calculating direct trust is to simply send an "ack" packet from the next hope node to the source node (Kaviani et al., 2021; Johnston et al., 2018). This confirms the receipt of a packet and updates a trust value within the source node. This is a less resource intensive technique compared with the watchdog mechanism, however, receiving a confirmation that a packets has been successfully received does not rule out the possibility of a malicious and deceptive node. And, with every forwarded packet requiring a delivery confirmation, the ack technique greatly increases the overhead of the network. Furthermore, this technique suffers from the same limitations as the watchdog mechanism. Specifically, when the next-hop node moves out of range of the transmitting UAV, the acknowledgment packet cannot be received, potentially causing the packet to be incorrectly flagged as dropped.

2.9.1.3 Indirect Trust Computation

The direct trust value, T^D, provides first hand information to node$_j$ regarding the trustworthiness of node$_k$. However, as mentioned previously, indirect trust information can also be combined with direct trust to provide a more detailed trust value. While node$_j$ has a single direct trust value regarding node$_k$, recommendations can be acquired from all neighbouring nodes regarding the trustworthiness of node$_k$. Combining these recommendations produce an indirect trust value. The indirect trust value is provided by a system called the *recommendation system* and is also responsible for the communication of indirect trust values. A simple recommendation system is proposed by Shabut et al. (2015), where second-hand experiences by neighbouring nodes are combined in the same way as the forward (f_{jk}) and drop (d_{jk}) packet counts in equation 2.4, with α' indicating positive indirect interactions and β' indicating negative indirect interactions, producing an indirect trust value, $T^{\mathrm{IN}} = \mathbb{E}[beta(\alpha', \beta')]$. Another way of combining indirect trust is by simply averaging the communicated direct trust values of other drones, like the model proposed by Tropea et al. (2024).

In contrast, some works only communicate negative recommendations, like the work of Li and Singhal (2007), which directly influences the overall trust value. On the other hand, some systems employ advanced mechanisms, like Dempster-Shafer theory (DST) (Wei et al., 2014; Yang et al., 2014). DST is a decision making mathematical framework

used in scenarios where the evidence obtained has a degree of uncertainty. The main feature of the DST protocol is the intelligent combination of indirect trust values from multiple sources. This is ideal in FANET scenarios where node communication enables the combination of trust values, but the evidence gathered from neighbouring nodes have a variable degree of trustworthiness.

2.9.1.4 Trust Aggregation

Once the direct and indirect trust values have been computed, the *overall trust* value, T^O, must be calculated using trust aggregation. There are many ways of combining direct and indirect trust, but one of the simplest ways is to use a weighted sum of direct trust, T^D, and indirect trust, T^{IN}, like in the work of Shabut et al. (2015); Wei et al. (2014):

$$T^O = \omega T^D + (1 - \omega) T^{\text{IN}}. \tag{2.6}$$

Where ω is the direct trust weighting $\in [0, 1]$. Once T^O is known, it is compared with a trust threshold, g. Nodes with $T^O \geq g$ are considered trustworthy and those with $T^O < g$ are considered malicious. This thesis refers to systems that use both direct and indirect trust as *direct-indirect aggregated trust mechanisms*.

2.9.2 Game Theory

Game theory is a mathematical framework for modelling situations in which multiple decision-makers, known as players, interact and make decisions that influence one another's outcomes (Fudenberg and Tirole, 1991). Each player selects a *strategy* with the goal of maximizing a numerical value representing the outcome or benefit they receive. This numerical value is referred to as a *payoff* and is calculated by use of a *utility function*. Core to the concept of game theory is the assumption that each player in the game is *rational*. In this context, rationality means that each player will make decisions that they believe will maximize their own gain, based on their preferences and the expectations about other players' decisions. Each player's decision impacts not only their own outcome but also the outcomes of other players, making game theory particularly relevant for fields that involve adversarial interactions, such as cybersecurity.

In FANET security, game-theoretic models are used to formalize interactions between nodes, helping to design intelligent response mechanisms. Some games model the uncertainty regarding a node's maliciousness by assigning utility functions to actions in the context of packet routing (Subba et al., 2018; Shila and Anjali, 2008). This enables nodes to make calculated forwarding decisions by considering all of the potential moves of the attacker. In contrast, other works define a game based on the interaction between a known malicious node and a benign node, where the benign node can choose

to monitor or not, and the attacking node can choose to launch an attack or remain inactive (Subba et al., 2018; Vijayalakshmi et al., 2023). The resulting payoff structure helps determine optimal monitoring strategies and resource allocation in adversarial environments. The analysis of these games can then be used to select the strategy which enhances the security or QoS of the FANET.

Subba et al. (2018) propose a system to protect VANETs against grey hole, black hole, modification and wormhole attacks. They use a cluster intrusion detection system (CIDS) module. In this module, Game theory is used here to analyse the trade-off between monitoring overhead and security. Here, the attacker node is modelled as a player who can attack the network or wait and the defender node has the option either to monitor the node or not. This is essentially a game of "chicken", where the attacker attempts to disrupt the network without being identified and the defender wishes to identify the attacker without affecting the QoS of the network.

Whereas Subba et al. (2018) focuses on improving network QoS, Lawrence and Latha (2016) applies game theory to infer whether a neighbouring node is benign or malicious. Incorporating this uncertainty into game theoretic models enhances their realism in practical scenarios. Moreover, another practical consideration is the trade off between packet forwarding and energy consumption. Specifically, the game theoretic model presented by Shila and Anjali (2008) describes a situation where a sending node is faced with the decision of sending a packet via an intermediate node (which may drop the packet) or directly to the destination node itself. Sending the packet to the destination node incurs a higher energy consumption, but is reliable. Alternatively, sending the packet via the intermediate node uses less energy but is not as reliable. This model is made more complex by the introduction of states which indicate the buffer of the sending node and the drop buffer of the intermediate node.

2.9.3 Reinforcement Learning

With the recent advancements in machine learning, it is no surprise that Reinforcement Learning (RL) (Sutton and Barto, 2018) has been used to create models to identify malicious nodes. Reinforcement learning is a machine learning framework which trains an *agent* to execute *actions* in specific *states* in order to increase a numerical reward signal. At first, the agent knows nothing about the environment in which it is placed. So, the agent trains itself by exploring its environment and discovering areas of the environment which yield the best reward.

Reinforcement learning aims to train a policy that tells an agent which action to take to earn more reward. In temporal-difference (TD) learning, the agent learns from trial and error: after each action it compares what it expected with what actually happened and nudges its estimates accordingly. Q-learning (Watkins, 1989) applies this idea by

estimating, for each state–action pair, how good that action is if the agent then behaves optimally. The agent prefers actions with higher Q-values because they promise greater long-term reward. Since Q-value estimates are unreliable early in training, the agent must initially explore unfamiliar actions before gradually shifting to the best action. One simple simple way of accomplishing this is to use the ϵ-greedy rule, where the agent explores by selecting a random action with probability ϵ, and exploits by selecting the action with the highest estimated Q-value with probability $1 - \epsilon$. As experience accumulates and estimates are updated, the Q-values stabilise and the policy approaches optimality.

Unsurprisingly, Q-learning has been integrated into numerous FANET protocols, both as a way of optimising routes through the network and as a defensive mechanism. Q-Routing, introduced by Boyan and Littman· (1993), uses Q-learning to discover optimal routes within ad-hoc networks. Q-routing has a relatively simple design, the state space represents the destination node while an action represents the selection of a next hop node. This makes both the state and action spaces the set of all node in the network (although, in reality the set of actions is defined by the set of current neighbouring nodes). The reward function is defined as the total delay in sending a particular packet, which includes the queue time and send time. In contrast to traditional Q-learning techniques, the Q-values in this model need to be minimised in order to find the optimal node selection with the smallest delay.

Q-learning based routing protocols have integrated network metrics to improve the QoS of the FANET under normal conditions. For example, the QTAR algorithm by Arafat and Moh (2022) uses node velocity and the residual energy of the next hop node node as a reward value to determine the optimal next hop node. In addition, QTAR uses an adaptive learning rate which scales depending on the two hop node delay. This means that the Q-values are updated in larger steps if the links are less reliable. Furthermore, the discount factor is also scaled according to the differences in the number of neighbouring nodes between time steps, i.e. a smaller discount factor indicates that the node's neighbours have changed significantly since the previous time step. The smaller discount factor makes the agent more myopic, prioritising shorter term QoS improvements. Similarly, Rahul and Kaarthick (2023) also integrate network performance metrics into their Q-learning algorithm, however, they use the network throughput to scale the learning rate. This accelerates learning when the throughput is low, encouraging a route to form faster.

Q-learning based routing protocols have also integrated defensive measures to assess the reputation of a node as well as its network state. For example, Q-Routing was later extended by Ryu and Kim (2023), who incorporate a trust value into their reward function, encouraging the node to send packets to trustworthy neighbouring nodes. They tested their enhanced protocol against grey and black hole attacks and proved that their technique performed significantly better than Q-routing.

2.9.4 Deep Learning and Other Machine Learning Methods

The recent interest in deep learning and other machine learning techniques has encouraged the integration of these methods into FANET defence solutions. Specifically, these models have demonstrated significant talent at amalgamating data from different parts of the system for enhanced decision making.

For example, Sbai and Elboukhari (2022) proposes an intrusion detection system which uses network performance statistics such as average round trip time, packet loss percentage and the number of packets received to identify grey hole nodes, black hole nodes, selfish nodes and benign nodes. A variety of machine learning models were trained, including a random forest classifier and a support vector machine.

Deep learning models have also been proposed for the classification of FANET threats, due to their robustness to noise, privacy preserving qualities and their ability to offer a degree of classification confidence. Meddeb et al. (2023) use semi-supervised learning techniques and auto encoders to generate class labels for unlabelled data. Then, a classifier is trained on this dataset to recognise Black Hole, Grey Hole and Flooding Attacks. Similarly, Hanafi et al. (2023) uses deep belief networks and long-short term memory models to recognise a variety of privileged escalation and Modification attacks. Simpler models were proposed by Almomani et al. (2016), which use feed forward neural networks to recognise flooding, scheduling, black hole and grey hole attacks.

While many deep learning techniques exist, those that better match the unique demands of FANETs are of particular interest. For example, Ceviz et al. (2025a) propose a federated learning based approach, FL-IDS, where nodes train an IDS locally and upload their model weights to a central location. Model weights are then aggregated and a new model is uploaded to all other UAVs in the network. This method is a particularly good choice for privacy conscious applications, as the training data transferred from UAVs to the GCS can be sniffed by malicious actors. By transferring model weights instead of data, malicious actors have a restricted view of the FANET. This model was later expanded upon by Ceviz et al. (2025b), who propose FSFL-IDS, which incorporates few-shot learning to address the scarcity of labelled samples. In their approach, each UAV has access to a support set consisting of a limited number of labelled data points (few-shot). Unlabelled data collected locally is then assigned class probabilities and incorporated into the training dataset of the IDS model.

So far, the models described in this section are designed to produce a crisp and actionable classification output. However, there may be some situations where capturing classification uncertainty may be useful. For example, Singh and Verma (2020) uses fuzzy logic to identify a node as bad, neutral or good. The input parameters to the fuzzy logic model is signal strength, transmission delay, node's energy, PDR, and past interactions. Likewise, Singh et al. (2020) implements a similar system which

produces a trust value as output. This value is compared against a minimum threshold, and nodes falling below the threshold are excluded from the network.

2.9.5 Summary of Related Work

Table 2.4 summarises the related work on protocols and IDS models. This table also includes other works which do not explicitly defend against network-layer cyber attacks, but do present interesting qualities which may assist in defending against some threats.

Work	IDS or Protocol	Defense Mechanisms	Defends Against
QTAR by Arafat and Moh (2022)	Protocol	Reinforcement Learning	None Specified
RORQ by Ryu and Kim (2023)	Protocol	Reinforcement Learning and Trust Management	Black Hole and Grey Hole Attack
RLTM by Jinarajadasa et al. (2018)	Protocol	Reinforcement Learning	None Specified
CBS-OLSR by Rahul and Kaarthick (2023)	Protocol	Reinforcement Learning	None Specified
SecGPSR by Tropea et al. (2024)	Protocol	Cryptography and Trust Management.	Grey Hole and Modification Attacks
Chawhan et al. (2022)	Protocol	Trust Management	Grey Hole Attack
TBCS by Singh and Verma (2020)	Protocol	Fuzzy Logic and Trust Management	Grey hole, Worm Hole and Sink Hole attacks
UNION by Barka et al. (2018)	Protocol	Trust Management	Black Hole, Modification and Worm Hole Attacks
Sadayan and Ramaiah (2022)	Protocol	Fuzzy Logic and Trust Management	Grey Hole, Black Hole and Modification Attacks
Subba et al. (2018)	Protocol	Game Theory	Grey Hole, Black Hole, Worm Hole and Modification Attacks
Shila and Anjali (2008)	Protocol	Game Theory	Grey Hole Attack
Ceviz et al. (2025b)	IDS	Few-shot and Federated Learning	Sink Hole, Modification and Black Hole Attack
Ceviz et al. (2025a)	IDS	Federated Learning	Sink Hole, Modification and Black Hole Attacks
Hanafi et al. (2023)	IDS	Deep Belief and Long Short-Term Memory Networks	Modification Attack (Among Others)
Almomani et al. (2016)	IDS	Machine Learning	Modification, Grey Hole and Black Hole Attacks
Bouhamed et al. (2021)	IDS	Reinforcement Learning	Modification Attack (Among Others)

TABLE 2.4: A Summary of popular protocols and intrusion detection models for ad hoc networks, indicating which network-layer cyber attacks they address (if any).

Chapter 3

System Model

In this section, we justify and construct our system model based on established research and methodologies within a search and rescue scenario. This system model is then translated into a set of simulation parameters and implemented within an NS-3 simulation. These simulations provide evidence to support the findings presented in all subsequent chapters of this thesis. Small deviations from the system model or threat model will be clarified in each separate chapter, if applicable.

Section 3.1 describes a typical SAR scenario with Sections 3.2 and 3.3 describing the architecture and UAV Specifications which were selected to model this kind of SAR scenario. The threat model is then presented in Section 3.4 which includes a detailed descriptions and algorithms of the grey hole attacks which the FANET is being exposed to. Finally Section 3.5 summarises the simulation parameters which are used throughout this thesis.

3.1 Application Scenario

In our work we focus on the use of UAVs and FANETs for search and rescue (SAR) applications. We substantiate our flying ad-hoc network (FANET) environment by describing a realistic, urban search and rescue (SAR) scenario based on a post-earthquake natural disaster. This earthquake has endangered human life and, in response, a search and rescue team has been deployed. The team has numerous tasks to complete, such as locating survivors, evaluate building structure and transporting supplies. Given the unfavourable environment conditions, these tasks come with considerable risk if human SAR teams were dispatched, therefore, UAVs are deployed in their stead. As the aforementioned SAR tasks require different sensors and equipment, three types of UAVs (described in Section 2.2), with appropriate devices, are dispatched simultaneously and form a FANET. Speed is a crucial aspect of any SAR

mission, and therefore, multiple drones are dispatched at once to accomplish these tasks in parallel. The FANET is controlled by a ground crew who operate from a GCS and once instructions are given to the UAVs, they are expected to be behave autonomously (Apvrille et al., 2014). However, some SAR tasks cannot be carried out by the drones independently. For example, voice communication between the ground crew and victims will need the support of the entire FANET to relay information back to the GCS (Mayor et al., 2019).

3.2 FANET Architecture

The FANET network presented in this section has a UAV ad-hoc architecture, depicted in Figure 2.2b, as it allows the base station to communicate with UAVs beyond the base station's normal communication range. This architecture has shown to be particularly useful for SAR scenarios (Bujari et al., 2017). In some cases, FANETs may use satellites in the place of GCSs, however, due to the limited number of satellites, access is limited and could be revoked at any time (The Washington Post, 2023). For this reason, GCSs are seen as a more reliable choice. The GCS is represented as a static node and another static node represents the UAV residing at the victim's location.

We assume that the FANET has already been fully deployed and is currently in active service. This means that the mobile nodes have been assigned their SAR missions and are moving to complete them. A proportion of the mobile nodes are designated as attacker nodes. In some experiments these nodes execute grey-hole attacks, while in control experiments the same nodes operate benignly using standard ad-hoc protocols. The UAVs are separated into three distinct groups: mobile malicious nodes, mobile defending (or benign) nodes and static nodes. Static nodes can only be benign and do not move. We classify UAVs by intended behaviour but do not assign fixed routing roles (for example, backbone vs. normal). This ensures every node can communicate with any other, increasing redundancy if a UAV fails.

The GCS and static node are placed at either end of a 1050m x 150m simulation area. The simulation area is split vertically into three sub-areas of equal sizes where each sub-area divides the number of attacking and benign nodes equally. Hence, the number of mobile UAVs in our simulation is a multiple of 3. As would be the case in real world scenarios, this FANET is designed such that the origin and destination node would not necessarily have a routing path at all times. To achieve this, the FANET consists of 24 mobile UAVs to encourage path sparsity. In addition, all mobile nodes are strictly bound to a specific sub-area, however, they are allowed to move freely within their designated area to accomplish their specific tasks. This setup, similar to previous works (Bansal and Baker, 2003; Leonov and Litvinov, 2018), ensures that data originating from the GCS traverses to the other side of the network via multi-hop routes consisting of

two or more forwarding nodes, which encourages multi-hop communication routes. For increased realism, data packets are also transferred in the other direction (static node to GCS) to simulate a voice call. This architectural setup is depicted in Figure 3.1.

In this thesis, we refer to specific *environments*, defined by two key parameters of the FANET: the attack node ratio (ANR) and the maximum node speed. The ANR denotes the proportion of attacking nodes relative to the total number of mobile nodes in the simulation, while the maximum node speed is the speed limit of the nodes within the FANET. There are a wide range of environment parameters which could be used to describe an environment, such as the choice of propagation loss model or mobility model. However, it has been noted that both ANR and maximum node speed are the most influential parameters during the evaluation of ad-hoc networks (Jari et al., 2021; Wei et al., 2014; Sánchez-Casado et al., 2015; Sadayan and Ramaiah, 2022; Singh et al., 2020; Bounouni et al., 2022; Rani et al., 2020; Ullah and Das, 2018), due to the fact that they make an exceptionally larger difference to the performance of the network compared with other environment parameters. Each environment is represented by a short form name to identify the environment more clearly. Specifically, the short form name takes on the format **"ANR"** followed by: the attack node ratio (ANR), **"Speed"**, and the speed in m/s. So, an environment with 6 malicious nodes and 18 benign nodes which all have a restricted max speed of 35 m/s would be indicated in its short form as: **ANR-25-Speed-35**.

In addition, we model the FANET in a 2D plane rather than a full 3D environment. While UAVs naturally operate in three-dimensional space, introducing altitude as an additional coordinate does not fundamentally alter the network dynamics under consideration, as connectivity in ad-hoc wireless networks is primarily governed by inter-node distance rather than height variation. Moving to a full 3D simulation would only become meaningful if we also modelled physical structures, terrain, occlusion, or altitude-dependent propagation effects. Such features require specialised simulation frameworks and significantly more complex environmental modelling, which lies beyond the scope of this study. By adopting a 2D representation, we capture the core spatial relationships of node density, horizontal separation, and multi-hop connectivity while maintaining a tractable simulation environment consistent with the objectives of our analysis.

3.3 UAV Specification

In order to ensure that our simulation environment accurately represents realistic UAV behaviour and communication dynamics in search and rescue (SAR) operations, this section defines the key parameters governing UAV mobility, routing, and communication capabilities. Section 3.3.1 outlines the chosen *mobility model* and

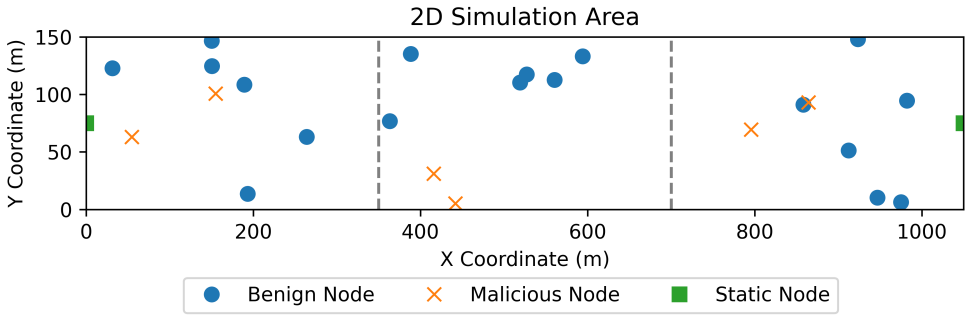


FIGURE 3.1: This Figure shows an example simulation environment with an ANR of 25%. The dotted lines represent the mobile node boundaries.

operational constraints such as speed and flight time, which are designed to reflect the autonomous nature and physical limitations of UAVs used in SAR missions. Section 3.3.2 describes the underlying *communication protocol*, which provides the routing foundation for UAV-to-UAV and UAV-to-ground control station (GCS) communication. Finally, Section 3.3.3 discusses the *communication range* adopted in our simulations, balancing realism with prior work to reflect the practical limits of UAV connectivity under non-line-of-sight (non-LoS) conditions.

3.3.1 Mobility

We have chosen to implement the Random Way Point (RWP) mobility model for the mobile nodes within our simulation due to its suitability for search and rescue scenarios (Alkhatieb et al., 2020; Tsao et al., 2022). From our application scenario, UAVs execute their SAR tasks autonomously, which result in mobility patterns that resemble the RWP mobility model. In addition, we choose a maximum speed range of 0-35m/s because of two reasons. Firstly, in contrast to the work of Srivastava and Prakash (2021) and Nayyar (2018), most studies use a maximum node speed range between 0 and 60 m/s (Mohammed Ahmed et al., 2021; Wheeb et al., 2023). This is speed range is revised further when considering that the UAVs reviewed in Section 2.2 have a maximum speed of 25 m/s. In conclusion, we decided to increase the maximum node speed to 35 m/s to include any other UAVs which have a slightly higher speed than the UAVs reviewed previously.

We have restricted our simulation time to fall in line with the minimum battery life of the UAVs in Section 2.2, as we do not consider recharge time within these simulations. The Inspired Flight IF1200 (Inspired Flight, 2024) has the minimum stated battery life of 24 minutes at its maximum take off weight. Therefore, we use 20 minutes as a conservative simulation time.

3.3.2 Protocol

In our case, all UAVs are able to communicate via the AODV (Perkins et al., 2003) protocol, described in Section 2.5. The AODV protocol is used for UAV-to-UAV communication as well as UAV-to-GCS and GCS-to-UAV communication. The AODV protocol is a popular routing protocol known for its low network overhead and its high throughput in FANET environments (Alkhatieb et al., 2020). We extend AODV with configurable defences designed to detect and mitigate grey-hole behaviour so the network can maintain forwarding performance under attack. These extensions are parametrised through the use of *configuration parameters*.

Given the target application of VoIP, we use UDP at the transport layer to take advantage of its connectionless design. Using a connectionless protocol means that the UAVs do not track packets or form sessions with the receiving device. By contrast, connection-oriented transport protocols (e.g. TCP) are ill-suited for FANETs, as their session management and stateful congestion control add overhead and react poorly to route churn.

3.3.3 Communication Range

The communication range of UAVs within simulated FANET environments is very broad, with some published work simulating communication ranges of 30m (Faraji-Biregani and Fotohi, 2021) to 1000m (Liu et al., 2023). Furthermore, depending on the category, the standards for 802.11 wireless communication enables a communication range between 80m and 1000m (Singh, 2016). For search and rescue operations, it is desirable for communication distances to be as large as possible to ensure adequate UAV coverage of the area (Gao et al., 2022). However, they are limited by, amongst other factors, line of sight (LoS). Studies with higher ranges do not take into consideration non-LoS communication distances, i.e. practical communication ranges of UAVs deployed in real life scenarios. The physical layout of the environment is inherently hard to predict and simulate, therefore, we provide a communication range which is both similar to previous works (Tropea et al., 2024; Sedjelmaci et al., 2021) and also reflects projections based on studies involving extended-range, non-LoS communication scenarios (Singh, 2016). In our simulation, we choose a maximum communication range of 380m.

3.4 Threat Model

At the start of the simulation, the malicious nodes are all assigned one specific grey hole attack class for the duration of the simulation. Similarly, defending nodes are assigned a

defence protocol. For experimental control, both attacking and defending nodes may also be assigned the benign class, enabling more consistent comparison across scenarios.

The attacking nodes are assumed to have full participation rights in the AODV routing protocol. They can generate and forward RREQ/RREP messages indistinguishably from other nodes, but they do not fabricate false routes (e.g. advertise an artificially low hop count). Their malicious influence is restricted solely to dropping data packets. Furthermore, the attacking nodes' physical mobility patterns are also identical to benign nodes, preventing easy behavioural detection. The attackers are assumed to possess only local knowledge of network topology, consistent with typical ad-hoc node behaviour. They do not have global situational awareness or coordination capabilities. Attacker nodes operate independently and do not share state, coordinate decisions, or collude.

In this section, we formalise the threat model that underpins our analysis by describing the operational context, adversarial behaviour, and defensive assumptions used throughout the thesis. Section 3.4.1 outlines the motivating scenario, including the attacker's objectives and the heterogeneous UAV environment in which the FANET operates. Section 3.4.2 then introduces the grey hole attack strategies considered in this work, distinguishing them from related packet-dropping attacks and defining the three specific variants evaluated in our simulations. Finally, Section 3.4.3 details the capabilities and limitations of the defending nodes, clarifying their monitoring scope, trust assumptions, and the defensive mechanisms available in different chapters of the thesis. Together, these subsections establish the behavioural assumptions and operational constraints necessary for the analyses that follow.

3.4.1 Scenario

The primary goal of the attacker is to degrade network availability while remaining undetected for as long as possible. Our motivating scenario from Section 3.1 states that the UAVs are heterogeneous, meaning that we can confidently assume that they are built by different manufacturers with different supply chains. With this in mind, we hypothesise that the network device within some of the UAVs has had a grey hole attack introduced at some point during the manufacturing process. Because these malicious components are effectively embedded into the platform before deployment, the operators have no practical means of verifying or removing them once the mission has begun.

Furthermore, in realistic FANET deployments, the communication protocol cannot simply be modified or upgraded mid-mission. Protocol updates require coordinated rollout, extensive re-testing, and controlled deployment environments that are incompatible with the real-time constraints of an active operation. Once the UAVs are

airborne, their software stack—including the routing protocol—must remain fixed to ensure stability and interoperability. As a result, the network operators must assume that any vulnerabilities present at launch will remain exploitable throughout the mission's duration.

This constraint is further exacerbated by the critical nature of many FANET applications, where mission failure can endanger both human life and high-value assets. Time-critical military, emergency-response, and search-and-rescue operations cannot afford the luxury of withdrawing compromised UAVs or replacing them with trusted alternatives during flight. In practice, once the fleet has been deployed, it must remain resilient to whatever combination of benign and compromised nodes is already present in the network. Defence must therefore occur at runtime, using only the capabilities available to the onboard networking stack and without altering the underlying protocol specification.

To counteract this threat under such strict operational constraints, the network personnel operating the FANET aim to strengthen the security of their routing behaviour against grey hole attacks by applying a variant of the AODV protocol. This defensive strategy must function within the fixed protocol architecture, requiring solutions that can detect or mitigate malicious behaviour during the mission without changing the fundamental routing framework or removing compromised UAVs from service.

3.4.2 Grey Hole Attack

In this thesis, we deliberately narrow the attacker model to focus exclusively on packet-dropping behaviour, omitting modification attacks, collaborative attacks, and other elevated threat capabilities in order to preserve realism, tractability, and relevance to the manufacturing-insertion scenario we consider. Modification attacks are excluded because they require adversaries to possess cryptographic keys, payload awareness, or the ability to manipulate and reconstruct protocol fields. Similarly, collaborative or colluding attacks are not considered, as compromised nodes introduced independently during manufacturing have no covert means to coordinate, share state, or synchronise their behaviour once deployed in a dynamic FANET.

From Section 2.8.2, black hole attacks and grey hole attacks behave very similarly. They both disrupt the network's availability by dropping packets, however, the mechanism by which they drop these packets are different. The *Black hole attack* uses the RREP/RREQ mechanism to attract traffic by advertising low hop routes. Once traffic has been attracted, the packets are subsequently dropped by the malicious node. This definition of a black hole attack seems to be universally accepted, however, there seems to be some variance in the definition of a *grey hole attack*. Some works define the grey

hole attack as a variation of the black hole attack, albeit, with a decreased packet drop rate (Mathonsi et al., 2023). In this thesis, we consider a grey hole attack which participates in all routing processes as if they were a legitimate node, making them exceptionally hard to identify (Singh et al., 2021).

To further exacerbate the situation, grey hole attacks also resemble benign packet dropping behaviour, making it yet more challenging to identify. Benign packet dropping behaviour is common in ad-hoc networking environments due to frequent topology changes as a result of node speed (Wu et al., 2013), packet collisions at the MAC layer (Maxa et al., 2017) and general congestion in the physical medium (Varadharajan and Balakrishnan, 2005).

To make the situation more challenging, we have given the malicious actor the choice of three types of grey hole attack. All of which are based on previous literature (Khanna and Sachdeva, 2019a; Jhaveri et al., 2012).

1. Firstly, the ***probabilistic*** grey hole attack drops packets probabilistically with probability $p \in [0, 1]$. This is assessed in every instance where a data packet is being forwarded by the grey hole node.
2. Secondly, The ***time-based*** grey hole attack drops all data packets within a specific time interval. For this attack, the simulation is divided up into time windows of length t_w. At the start of each time window, the variable, $b \in [0, 1]$, determines the probability that the time window will be a *drop window*. Within the drop window, if a request to forward a data packet is made, the data packet is dropped.
3. Thirdly, the ***precursor-based*** grey hole attack drops all data packets from a specific precursor node in the routing path. To explain further, every node in the network at the start of the simulation is assigned a *drop-node* status with probability $q \in [0, 1]$. The drop node status, which persists for the entire simulation, is a boolean variable that indicates that a specific node will have its data packets dropped by the grey hole node if it acts as a precursor. In addition, each attacker node will have a unique assignment of drop-node statuses and, as we assume there is no collusion between attacking nodes, the drop node status can be assigned to other attacking nodes.

We shall make reference to these specific types by the names indicated in bold italics above. Furthermore, the variables defined in the list above (p, b and q) act as the attack configuration parameters for each grey hole type and are generally refereed to as the protocol's intensity. Varying the intensity value allows the malicious actor to execute covert (lower intensity) or overt (higher intensity) attacks. Table 3.1 summarises the attacks used in this study, detailing their attack type, short form name and any associated variable(s). In addition, we also provide a more detailed description of the attacks in algorithmic form. Please see Algorithms 1, 2 and 3.

Algorithm 1 Probabilistic Grey Hole Attack

```
Input p                                              // Probability of packet drop

In Forwarding(recNode) Function:
if data packet received then
    p̂ ~ U(0,1)
    if p̂ < p then
        Drop packet
    else
        Forward packet
    end if
end if
```

Algorithm 2 Time-based Grey Hole Attack

```
Input b                                              // Probability of drop window
Input w                                              // Drop window length
Input t                                              // Current time of the simulation
Input maxSimTime                                     // Maximum simulation time
Input currentlyDropping    // Indicates if the node is currently dropping packets

In Timer() Function:                  // This timer function executes every second.
while t < maxSimTime do
    if t mod w is 0 then
        b̂ ~ U(0,1)
        if b̂ < b then
            currentlyDropping ← True
        else
            currentlyDropping ← False
        end if
    end if
end while

In Forwarding(recNode) Function:
if data packet received then
    if currentlyDropping then
        Drop packet
    else
        Forward packet
    end if
end if
```

TABLE 3.1: List of Attacks Used in this Thesis

Grey Hole Attack Type	Short Form	Variables
Probabilistic	**PD–0.2**	$p = 0.2$
Probabilistic	**PD–0.4**	$p = 0.4$
Probabilistic	**PD–0.6**	$p = 0.6$
Probabilistic	**PD–0.8**	$p = 0.8$
Probabilistic	**PD–1.0**	$p = 1.0$
Time-based	**Tdrp–0.3**	$b = 0.3$ $w = 5$ seconds
Time-based	**Tdrp–0.5**	$b = 0.5$ $w = 5$ seconds
Time-based	**Tdrp–0.7**	$b = 0.7$ $w = 5$ seconds
Precursor-based	**Sele–0.3**	$q = 0.3$
Precursor-based	**Sele–0.5**	$q = 0.5$
Precursor-based	**Sele–0.7**	$q = 0.7$

Algorithm 3 Precursor-based Grey Hole Attack

```
Input q                                   // Probability of drop-node status
Input isDropNode[]                        // The drop-node status of all nodes

At Start() only:
for each node do
    q̂ ~ U(0,1)
    if q̂ < q then
        isDropNode[node] ← True
    else
        isDropNode[node] ← False
    end if
end for

In Forwarding(recNode) Function:
if data packet received then
    if isDropNode[recNode] then
        Drop packet
    else
        Forward packet
    end if
end if
```

3.4.3 Defender Capabilities

Defender nodes have passive monitoring capabilities and can observe local forwarding behaviour, neighbour transmissions, and their own packet delivery ratios. However, they do not possess global visibility of the network. Defenders assume the correctness of AODV control packets and do not employ cryptographic authentication in the baseline

model. However, they do not inherently trust any node's forwarding behaviour. The defence system is dependent on the specific model implemented, for example, Chapter 4 is detection-oriented only and does not implement automatic node isolation or route blacklisting. In contrast, in Chapter 6, defenders are free to block specific nodes from the network using appropriate protocol adjustments that does not affect other routes.

One method of defending against packet dropping attacks is to grant global network knowledge, however, such visibility is fundamentally infeasible in highly mobile, bandwidth-constrained ad-hoc environments, where maintaining a consistent global view would impose substantial communication overhead and contradict the decentralised nature of FANET operation.

3.5 Simulation

This section details the simulation setup including the selection of the network simulation software in Section 3.5.1

3.5.1 Software Selection

Several established network simulators are used in wireless and ad hoc network research. Dorathy and Chandrasekaran (2018) identify NS-3, NS-2, OMNeT++, OPNET, QualNet, NetSim, JiST, J-SIM and GloMoSim as the primary tools available for wireless network evaluation. In order to select a simulator suitable for the needs of this research, the following key requirements were established.

First, the simulator needed to be in active development, ruling out older tools such as GloMoSim and JiST/SWANS, which have seen no modern updates and lack current documentation or support. Second, because this thesis builds upon the work of flying ad hoc networks, the simulator required robust mobility modelling and the ability to simulate dynamic multi-hop wireless topologies. Third, the protocol-level experiments required a network layer implementation of AODV that could be modified, enabling the protocol's forwarding logic and security-related behaviour to be altered directly. Finally, a Python–C++ execution environment was necessary in order to simulate networking components and train machine learning models concurrently.

NS-3 met all of these requirements. It provides an actively maintained C++ codebase, extensive documentation, and a realistic representation of ad hoc environments (Avneet et al., 2017). It is widely used in the field of ad hoc security research (Sedjelmaci et al., 2021; Subba et al., 2018; Tang et al., 2023), making it a credible and well-validated choice. Crucially, the ns3-gym framework enabled parallel execution of the simulation environment (C++) and AI modules (Python), allowing the simultaneous training of

TABLE 3.2: Simulation Parameters

Parameter	Value
Simulation Time	1200 Seconds
Mobile Benign UAVs	16, 18, 21
Mobile Malicious UAVs	3, 6, 9
Total Mobile Nodes	24
Static UAV	1
GCS Station	1
Simulation Area	1050m x 150m
Mobility Model	Random Way Point
UAV Communication Range	380m
Propagation Delay Model	Constant Speed
Propagation Loss Model	Friis
Packet Size	1024 Bytes
Packet Tx Rate	10 pkts/sec
Traffic Type	UDP

models and network simulation. While OMNeT++ offers a Python extension, it does not support this kind of parallel execution, and OPNET lacks sufficient wireless mobility support to be suitable for airborne or highly dynamic ad hoc networks. For these reasons, NS-3 combined with ns3-gym was the only simulator that satisfied all functional, experimental and developmental requirements.

3.5.2 Implementation

Building upon sections 3.5.1, we build an NS3 simulation environment with the simulation parameters detailed in Table 3.2. These system model design decisions correlate with other recent studies which investigate the performance of routing protocols (Alkhatieb et al., 2020). The simulation can be executed multiple times with different seeds which randomises the initial positions and velocities of nodes to ensure diverse testing scenarios.

Chapter 4

A FANET Dataset for Fast Classification of GHAs

This chapter examines the temporal characteristics of GHAs in FANETs, exploring whether their behaviours exhibit identifiable patterns over time. By representing node interactions as multivariate time series, the chapter frames threat detection as a classification problem and evaluates the feasibility of making early predictions without observing entire sequences. To support this, it introduces a new synthetic dataset of UAV interactions and validates its utility through an early time series classification model. The findings demonstrate that both the type and intensity of GHAs can be distinguished at different stages of communication.

Section 4.1 introduces and motivates the creation of a new dataset, while Section 4.2 describes how the dataset is formed. To evaluate the dataset, we create an early time series classification model and train it using our dataset. Section 4.4 presents the results and finally, Section 4.5 concludes.

4.1 Introduction

Despite the development of FANET IDSs (presented in Section 2.9) and more general IDSs designed for the early detection of cyber threats (Thwaini, 2022; Maddireddy and Maddireddy, 2020), there are a lack of FANET IDSs which optimise for both fast threat detection as well as accurate classification. This specification led us to a body of literature called *early time series classification* (ETSC) which has been successfully applied in other safety critical systems, such as gas leak detection (Hatami and Chira, 2013) and blood purification treatment (Ghalwash et al., 2013). In addition, the importance of accuracy and decision latency is also evident in the cyber security field. Specifically, fast classification models are utilised in malware identification by analysing application

programming interface (API) call sequences (Sharma and Kumar Singh, 2021) and in the hardware security domain (Woo et al., 2018) by analysing hardware performance counters.

Training and evaluating an ETSC model for grey hole attack detection in FANETs requires a suitable dataset. Section 4.1.1 describes the disadvantages of current datasets while Section 4.1.2 describes the solutions which address these disadvantages. Finally, Section 4.1.3 presents a recap of the main factors of each dataset discussed and how FAN-GHETS24 differentiates itself from these datasets.

4.1.1 Motivation

Training and evaluating an ETSC model for GHA detection in FANETs requires a suitable dataset. Consequently, we have identified 3 essential properties that must be present to ensure its versatility compared with other datasets in the field. Firstly, the differences in the test-bed platforms used to generate some network security datasets are not compatible with the structure of FANETs. Secondly, the structure of previous datasets are not conducive when training fast time-series classification models. And thirdly, previous datasets do not include the range of grey hole attacks which are considered in this thesis. We describe these requirements in more detail in subsections 4.1.1.1, 4.1.1.2 and 4.1.1.3.

4.1.1.1 Unsuitable Environment

The dataset needs to be generated from a platform which correctly simulates FANETs and the affects of GHA nodes on the FANET. There are several prominent datasets in the network security domain, which include the KDDcup'99 (Stolfo Salvatore and Chan, 1999), NSL-KDD (Tavallaee et al., 2009), and the CIC-IDS2017 (Sharafaldin et al., 2018) datasets. These datasets are still used to develop IDS models today (Khatib Sulaiman Dalam No et al., 2024; Jose and Jose, 2023; Vibhute et al., 2024), however, they are of limited use in FANET environments due to their network configuration. The KDDcup'99 (Stolfo Salvatore and Chan, 1999), NSL-KDD (Tavallaee et al., 2009) and CIC-IDS2017 (Sharafaldin et al., 2018) datasets are generated using simulations of local area networks (LANs) where routing is accomplished through dedicated switches and routers. In contrast, the FANET described in Chapter 3 uses a type of ad-hoc networking where the nodes of the network can simultaneously function as senders, receivers, and routers. In addition, the KDDcup'99 (Stolfo Salvatore and Chan, 1999), NSL-KDD (Tavallaee et al., 2009) and CIC-IDS2017 (Sharafaldin et al., 2018) datasets do not incorporate node mobility, a crucial factor influencing the performance of FANETs.

4.1.1.2 Dataset Construction

FANET datasets for ETSC require a slightly different construction compared with other FANET datasets (Zhai et al., 2023; Ceviz et al., 2025b; Almomani et al., 2016). ETSC datasets require a label to be assigned to an ordered sequence of events, rather than a single data point. This differs from other FANET datasets, where labels are typically applied to averages or counts of metrics over fixed time periods (Almomani et al., 2016; Ceviz et al., 2025b). For example, the dataset proposed by Ceviz et al. (2025b) provides a count of the number of forwarded route request packets in 5 second intervals (Sen and Clark, 2011). Conversely, the dataset proposed by Zhai et al. (2023) assigned labels to individual packets or metrics instead of providing averaged values. However, this method does not preserve event ordering, limiting its suitability for ETSC models.

4.1.1.3 Grey Hole Attack Intensities and Types

In contrast to previous work, the dataset proposed by Almomani et al. (2016) incorporates a GHA. However, this dataset did not simulate varying types or intensities of GHAs, despite evidence from previous research showing that both factors can significantly affect the selection of defence strategies (Bounouni et al., 2022; Tropea et al., 2024).

4.1.2 Contributions

To address the issues presented in Section 4.1.1, This chapter introduces FAN-GHETS24, the **FAN**et **G**rey **H**ole dataset for **E**arly **T**ime **S**eries classification. Specifically, FAN-GHETS24 has three properties which are realised through three key contributions that define the design of our dataset. Firstly, the FAN-GHETS24 dataset uses the system model from Chapter 3 to simulate the mobility and low level networking components of the FANET environment. This ensures that the FAN-GHETS24 dataset is suitable for FANETs.

Secondly, we construct the dataset using sequences of packet-level interactions between pairs of nodes, where each sequence is labelled according to the behaviour of the node being interacted with. This structure captures temporal patterns essential for training ETSC models. These sequences undergo post-processing via two methods: firstly, an anonymization procedure that replaces IP addresses with standard string variables, allowing for offline model training and universal deployment across UAVs; and secondly, the application of feature engineering techniques to format the data for machine learning model integration.

Thirdly, the dataset includes all the GHAs defined in our system model, offering a broader range of attack scenarios compared to previous datasets (Almomani et al., 2016).

In addition, this chapter also introduces the ETSC model, CEC-FANET, the **C**onfidence-based **E**arly **C**lassification model for **F**lying **A**d-hoc **NET**works. CEC-FANET is trained on sequences from the FAN-GHETS24 dataset and uses the combined grey hole attack intensity and type as a label. To quantify the rapidity of classification, a Decision Latency score measures the proportion of the time series observed before a decision is made, expressed as a percentage of the total sequence length. Additionally, the accuracy of the model is defined as the number of correctly classified sequences as a proportion of the total number of sequences. This model achieved an accuracy of 70% with an average Decision Latency of 62%. By comparison, a fixed time series classifier, which uses 63% of the sequence length, achieved only 65% accuracy. This indicates a 5% improvement in accuracy when using an ETSC model.

4.1.3 Summary

Table 4.1 summaries the datasets we have reviewed and indicates the attractive properties of FAN-GHETS24. For clarity, dataset names which are in italics are "given names", that is, a name which has been assigned to that dataset in the absence of any other name.

TABLE 4.1: Comparison of Datasets

Dataset Name	Suitable for FANET Environments	Suitable for ETSC Models	Grey Hole Attack Type and Intensity Included	Publicly Available
KDDcup'99 (Stolfo Salvatore and Chan, 1999)	✗	✓	✗	✓
NSL-KDD (Tavallaee et al., 2009)	✗	✓	✗	✓
CIC-IDS2017 (Sharafaldin et al., 2018)	✗	✓	✗	✓
WSN-DS (Almomani et al., 2016)	✗	✗	✗	✓
FL-IDSDS (Ceviz et al., 2025b)	✓	✗	✗	✗
ETD-DS (Zhai et al., 2023)	✓	✗	✗	✗
FAN-GHETS24*	✓	✓	✓	✓

*Proposed Dataset

4.2 Dataset

FAN-GHETS24 consists of sequences of interactions between any benign node in the network, which we will refer to as the *eval node*, and any other node within the communication range of the eval node which is the subject of the evaluation, referred to as the *subject node*. The eval node investigates the subject node by collecting their packet interactions and any promiscuous information related to the subject node. Through these interactions, the eval node can gradually accumulate evidence to classify the subject node into a particular category.

Using the system model from Chapter 3, simulations are executed with an information logging module, introduced in Section 4.2.1 which is used to collect packet information exchange from the perspective of the eval node. We filter this information by examining the interaction between this node and a subject node, of which, the details of this examination are presented in Section 4.2.2. This interaction data is in its raw form, meaning that the data contains IP addresses which uniquely identify the node. IP addresses are likely to differ in subsequent simulation runs, therefore, training on this data would be counter productive. To remedy this situation, we also introduce an anonymization step in Section 4.2.2, which learns the characteristics of the node without learning specific IP addresses. In addition, the next post-processing step, outlined in Section 4.2.3, introduces feature engineering techniques designed to enhance the dataset's suitability for training fast classification models.

4.2.1 Information Logging

Packet information is collected at the network layer through two routes: 1) through the forwarding function of the protocol and 2) via promiscuous mode (See Section 2.9.1.1). Only benign nodes collect traffic from the network layer, as malicious nodes do not classify other nodes in the network and therefore have no incentive to collect such data. If a packet is received via promiscuous mode, we say it is *witnessed* by a particular node and is recorded in the log file. To clarify, we define a *record* as a single packet sent, received, or witnessed that contains 13 metrics related to that particular packet. These 13 metrics are summarised in Table 4.3. Of these 13 metrics, mobility metrics are also collected alongside the sending and receiving of packets.

Specifically, Every UAV in the simulation has the ability to measure their own speed and position, as well as the positions and speeds of neighbouring nodes. Similar to a previous work by Arafat and Moh (2022), this information can be summarised in two forms: *distance* and *direction*. The distance, $\delta_{ij}(t_k)$, is defined as the euclidean distance between node$_i$, and a node$_j$ at time t_k. Moreover, we use relative distance

measurements between time steps, to define direction:

$$\psi_{ij}(t_k) = \delta_{ij}(t_k) - \delta_{ij}(t_{k-1}), \tag{4.1}$$

which indicates if the two nodes are gradually moving away or moving towards each other.

A custom AODV module allows the logging of these packet metrics. Moreover, due to the availability of routing tables and mobility models, the packet metrics are fused with the information available from these sources. As mentioned in Section 2.4, when data packets are routed via AODV, only the IP address of the destination node is present in the IP header, as information regarding the next hop node is stored in the routing tables of the intermediate nodes. Both the sender and receiver node have information regarding which node they need to forward to or receive from, which means that this information does not need to be stored in the header of the data packet. By using logging statements within the AODV routing module, we can extract these IP addresses by accessing the routing table.

The records within our simulation are timestamped, enabling the preservation of packet ordering. The AODV control messages are extracted to establish situational awareness regarding routes and broken connections. The IP addresses associated with all control and data packets are used to identify which nodes are sending and receiving data. These node IP addresses are stored as three main metrics:

- The **Seen By** IP address is the node which has sent, received or witnessed a packet.
- The **Sent By** IP address is the node which has sent this packet according to the "Seen by" IP address.
- The **Sent To** IP address is the node which the **Sent By** IP address has sent its packet to (from the point of view of the **Seen By** IP). The **Sent By** IP address can also be the broadcast address.

Mobility is an important metric for the identification of GHAs in FANET environments. Therefore, the directions and distances from the **Seen By** node to the **Sent To** and from the **Seen By** node to the **Sent By** node are recorded at the time the packet is sent, received or witnessed by the **Seen By** node. Of course, some of these packets are sent to a broadcast address and, as a node is not associated with this address, the direction and distance are set to 0m. Similarly, if the **Seen By** node is the same as the **Sent By** node, then the distance and direction are set to 0. There are also other situations where distance and direction need to be set manually. For example, if an IP address is out of range, then the distance is limited to the maximum communication range of the UAVs and the direction is set to 0m.

The metrics discussed so far provide the foundation for contextual awareness and node identification. To further support grey hole threat classification, we introduce two additional metrics: time-to-live (TTL) and packet ID. While their individual values carry little meaning, tracking the recurrence of packet IDs and the decrement of TTLs across a sequence exposes patterns in node interactions and potential threats. This approach is analogous to the watchdog mechanism described in Section 2.9.1.1, but unlike the watchdog, models that use FAN-GHETS24 sequences are not constrained by timeout parameters.

The attack AODV protocol was also modified to include its own logging procedure which records the timestamp of every packet dropped. This provides useful information when analysing the final results of an ETSC model.

4.2.2 Node Perspective and Anonymization

So far, we have described the data collection process that captures all node communications across the entire FANET. This raw data must be further processed to construct sequences that represent the complete history of interactions between specific eval–subject node pairs. To illustrate this process, we focus on a single eval node, which generates one or more sequences depending on the subject nodes it interacts with. The same procedure is repeated for each eval node in the FANET. It is important to emphasize that eval nodes are always benign, while subject nodes may be either benign or malicious. Accordingly, each sequence is labelled based on the class of the subject node. The processing steps are outlined below:

1. **Filter by Evaluation Node Perspective**: Firstly, the data is filtered so that the IP addresses in the **Seen By** column is the IP address of the eval node. The resultant data contains packet interactions which have been sent, received or witnessed by the eval node. The output of this process is known as the eval data. As each record in the Seen By field now contains the same value, this can be safely removed.
2. **Group by Contacted Subject Nodes**: The eval data is further segmented based on the set of nodes that the eval node has encountered during the simulation. Each of these contacted nodes becomes a distinct subject node. So, for each eval-subject node pair, we apply the following procedure:
   (a) **Sent To Filtering**: The eval data is filtered according to the Sent To field, where the Sent To IP address is equal to the IP address of the subject node, origin node, destination node or the broadcast address.
   (b) **Sent By Filtering**: Separately, the eval data is also filtered according to the Sent By field, where the Sent By IP address is equal to the IP address of the subject node, origin node or destination node.

IP Address (Example)	String Replacement	Description
10.1.1.8	MY_IP	An Eval Node IP
10.1.1.14	INSPECT_IP	A Subject Node IP (Benign)
10.1.1.26	STATIC_NODE_IP	The Static Node IP
10.1.1.25	GROUND_CONTROL_IP	The GCS Node IP
10.1.1.255	BROADCAST_IP	The Broadcast Address
10.1.1.9	OTHER_IP	A Benign Node's IP Address
10.1.1.22	OTHER_IP	A Malicious Node's IP Address

TABLE 4.2: This table describes the string values which are substituted for specific IP addresses during the anonymisation procedure. We give example IP addresses here to illustrate how these addresses are substituted. For example, in this instance, the subject node is a benign node and the other IP addresses (one benign and one malicious) have been substituted as "Other".

(c) **Concatenation**: The Sent By and Sent To filter data is then concatenated. We refer to to this output as the concatenated data.

(d) **Anonymisation**: We apply an anonymisation procedure which replaces IP addresses with string variables. Anonymizing the sequences means that they are IP address agnostic, therefore, any trained model can be integrated into any node within the FANET. Every IP address in the Sent To and Sent By fields of the concatenated data is substituted with "MY_IP", "INSPECT_IP", "STATIC_NODE_IP", "GROUND_CONTROL_IP", "BROADCAST_IP" or "OTHER_IP". "OTHER_IP" can be thought of as a catch-all string representation which substitutes any IP address if the IP address has not already been substituted by a previous string value. See Table 4.2 for an example of how IP addresses are substituted with string values.

(e) **Label Sequence by Subject Node Class**: This generated sequence is then labelled according to the class of the corresponding subject node.

While effective, this node perspective and anonymisation approach does introduce some limitations. For example, 3rd party information, i.e. information from other sequences, cannot be used to classify a particular subject node due to the anonymisation process. Moreover, the lack of routing context also means that packets cannot be tracked through the simulation. More practically, in real world deployments, a log would have to be continuously filtered, anonymised and stored by the UAV in order to feed the IDS system. If an eval UAV has many neighbouring UAVs, the filtering process could consume significant resources.

Extracted Data	Feature Engineering Technique	Brief Description
Timestamp	One-step difference & Normalisation	Time the packet was sent/received/witnessed; processed as elapsed time since previous packet.
Seen By IP address	Anonymization & Dummy Encoding	IP address of the node that sent/received/witnessed the packet.
Sent To IP address	Anonymization & Dummy Encoding	Destination or next-hop IP address for the packet.
Sent By IP Address	Anonymization & Dummy Encoding	IP address of the node that transmitted the packet.
Network Packet Type	Dummy Encoding	An enumerated type indicating if the packet is an RREP, RREQ, RERR or data packet.
Packet ID	Normalisation	Unique packet identifier (see Section 2.4).
TTL	Normalisation	Time-to-live field (see Section 2.4).
Is Promiscuous Mode	None	Indicates whether the packet was received via promiscuous mode.
Node's Current Speed	Normalisation	Current node speed in m/s.
Distance to **Sent To** IP Address	Normalisation	Euclidean distance to the **Sent To** node from the current node (0 if broadcast).
Direction to **Sent To** IP Address	Normalisation	Direction to the **Sent To** node (as defined in Equation 4.1); 0 if broadcast.
Distance to **Sent By** IP Address	Normalisation	Euclidean distance to the **Sent By** node (0 if node is itself).
Direction to **Sent By** IP Address	Normalisation	Direction to the **Sent By** node (as defined in Equation 4.1); 0 if node is itself.

TABLE 4.3: Extracted Data From Simulations

4.2.3 Feature Engineering

Sharma et al. (2024) show that utilising a machine learning based approach for early classification is effective when the training dataset is large. Hence, we expect the majority of researchers working with our dataset to use machine learning models. To aid these researchers, we enhance the suitability of our dataset for machine learning models by applying feature engineering techniques, as suggested by Zheng and Casari (2018). The metrics from our simulation encompass various data types, each

necessitating specific processing methods. Please refer to Table 4.3 for a description of the data processing methods used for each data field. In the remaining paragraphs of this section, we give more detail of why these feature engineering techniques are used. Many of these metrics, such as direction, distance, TTL and speed are simply normalised. In contrast, other feature engineering techniques require more in depth explanation.

We find that the time difference between timestamps can provide contextual meaning, as the inter-packet time delay can infer packet loss patterns over time or the freshness of routing paths. So, we include the time differences between packets as a feature within our dataset. We then normalise these timestamp differences to produce the final feature.

Due to the anonymization process, each IP address field now has a fixed number of potential values, therefore, we can implement dummy encoding, as suggested by Zheng and Casari (2018), for the **Sent To** and **Sent By**. Notice how the **Seen By** node is not included in the process, as once the anonymization process is completed, the IP address of the **Seen By** node does not add any information. Furthermore, we can also implement dummy encoding for the network packet type metric as that metric specifies if the packet is an AODV control message or a data packet. Overall, the implementation of dummy encoding means that the unique values in these fields are now represented by a sparse matrix of binary values. This is beneficial for machine learning algorithms, but also increases the dimensionality of the feature space.

4.2.4 Statistics

The FAN-GHETS24 dataset consists of 31910 sequences across 12 classes. The distribution of classes is shown in Figure 4.2, which predictably shows that the benign class is in the majority, as one would expect in cyber security scenarios. In addition, the distribution of sequence lengths in Figure 4.1 shows that the majority of sequences in FAN-GHETS24 have longer lengths than what is used in the validation model. This means that there is potential to build larger models capable of handling longer sequences, which could improve the performance of future ETSC algorithms.

With the feature engineering techniques implemented, a single data point or record in any sequence consists of 8 continuous features, represented by the float data type, and 17 categorical features, represented by the boolean data type. In total, there are 25 features per record. Table 4.4 shows 4 example raw outputs from the simulation which can be processed into records via the techniques described previously. Table 4.5 shows 4 example records from the dataset which have not been generated from the raw output in Table 4.4. For every sequence in the dataset, there are two labels: 1) the grey hole attack type and 2) the grey hole attack intensity. The labelling has been divided to enable different types of classification models to be trained. For example, one may wish

Metric	Raw Output 1	Raw Output 2	Raw Output 3	Raw Output 4
timestamp	1013304127	1013304165	1013304253	1013304263
seen_by	10.1.1.3	10.1.1.5	10.1.1.6	10.1.1.4
sent_to	10.1.1.255	10.1.1.255	10.1.1.255	10.1.1.255
sent_by	10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1
net_packet_type	2	2	2	2
packet_id	-1	-1	-1	-1
ttl	1	1	1	1
promisc_mode	1	1	1	1
current_speed	0.040265	0.496471	0.576345	0.24888
sent_to_d_distance	0	0	0	0
sent_to_distance	0	0	0	0
sent_by_d_distance	0.0203302	-0.217512	-0.222136	-0.0804193
sent_by_distance	0.0357778	0.0472725	0.0717229	0.0748423

TABLE 4.4: This table shows 4 records from the simulation output. These four packets are the result of a single broadcast message from 10.1.1.1, which was witnessed by four different nodes. The broadcast message is an AODV RREP message with a TTL of 1, indicating that this is a HELLO message.

to train a model to recognize only the type of grey hole attack, rather than both the type and intensity of the attack.

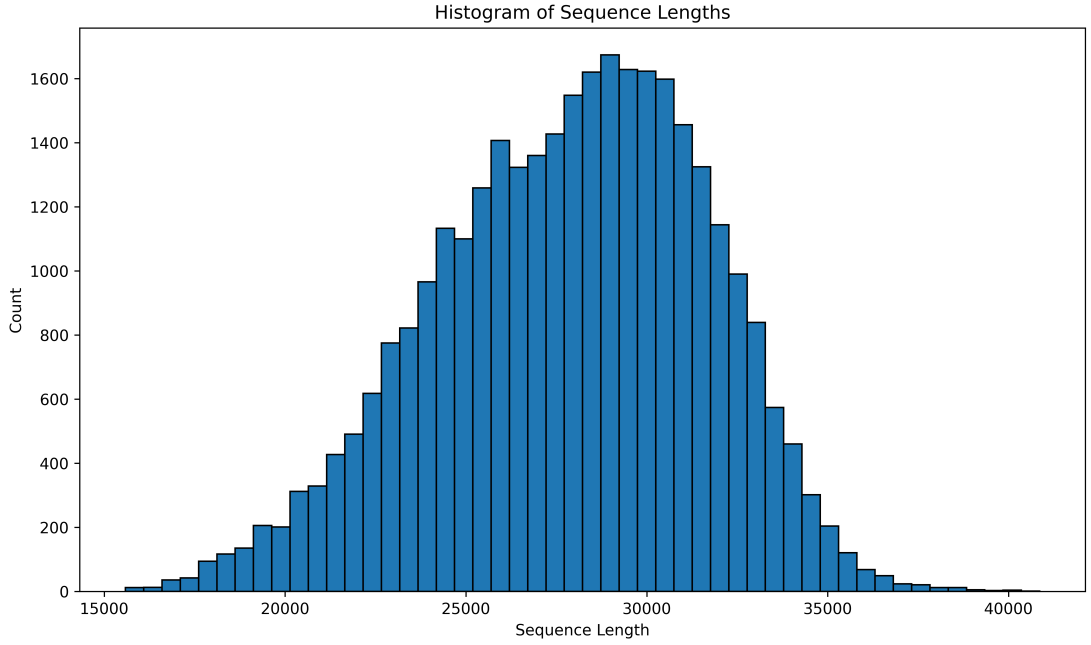


FIGURE 4.1: This graph shows a histogram of sequence lengths in FAN-GHETS24.

4.2.5 Usage

The FAN-GHETS24 dataset consists of sequentially numbered parquet files in the format: "example_host_data_$srID$.parquet", where $srID$ is the sequence reference ID.

Feature	**Type**	Record 1 ($X_{t,:}$)	Record 2 ($X_{(t+1),:}$)	Record 3 ($X_{(t+2),:}$)
net_packet_type_DATA_PACKET	Categorical	1.00000	1.00000	1.00000
net_packet_type_RERR_PACKET	Categorical	0.00000	0.00000	0.00000
net_packet_type_RREP_ACK_PACKET	Categorical	0.00000	0.00000	0.00000
net_packet_type_RREP_PACKET	Categorical	0.00000	0.00000	0.00000
net_packet_type_RREQ_PACKET	Categorical	0.00000	0.00000	0.00000
promisc_mode	Categorical	1.00000	1.00000	1.00000
sent_by_GCS_IP	Categorical	0.00000	0.00000	0.00000
sent_by_INSPECT_IP	Categorical	0.00000	0.00000	0.00000
sent_by_MY_IP	Categorical	0.00000	0.00000	0.00000
sent_by_OTHER_IP	Categorical	0.00000	0.00000	0.00000
sent_by_STATIC_NODE_IP	Categorical	1.00000	1.00000	1.00000
sent_to_BROADCAST_IP	Categorical	0.00000	0.00000	0.00000
sent_to_GCS_IP	Categorical	1.00000	1.00000	1.00000
sent_to_INSPECT_IP	Categorical	0.00000	0.00000	0.00000
sent_to_MY_IP	Categorical	0.00000	0.00000	0.00000
sent_to_OTHER_IP	Categorical	0.00000	0.00000	0.00000
sent_to_STATIC_NODE_IP	Categorical	0.00000	0.00000	0.00000
current_speed	Continuous	0.37413	0.37413	0.37413
packet_id	Continuous	0.27491	0.27501	0.27512
sent_by_d_distance	Continuous	0.40920	0.40920	0.40920
sent_by_distance	Continuous	0.62402	0.62402	0.62402
sent_to_d_distance	Continuous	0.50000	0.50000	0.50000
sent_to_distance	Continuous	1.00000	1.00000	1.00000
timestamp difference	Continuous	0.00127	0.00330	0.00233
ttl	Continuous	0.96923	0.96923	0.96923

TABLE 4.5: This table shows 3 example data records from one sequence. This specific sequence shows 3 data packets which have been sent from the static node to the GCS node. These packets have been witnessed via promiscuous mode, and, have been sent in rapid succession, as indicated the "timestamp difference" feature. This data is not the result of processing the simulation data from Table 4.4. Column notation will be described in a subsequent section.

Parquet files are a compressed format which are easily readable by the Pandas python library. Every parquet file is paired with a JSON file with the same sequence reference ID and has the file name format: "example_host_data_$srID$.json". This JSON file contains metadata regarding the parquet file of the same sequence ID. The metadata includes information regarding the total sequence length, the class which the subject node belongs to, a list of attack points (If applicable) and the feature data types. Attack points are specific time step values that indicate when a packet was dropped due to the presence of a GHA node.

For anyone wishing to recreate our dataset, we provide an explanation of the configuration files used for the simulation. The benign and attack configuration files are in the format "*.ini" and provide a list of benign or attack protocol parameters called

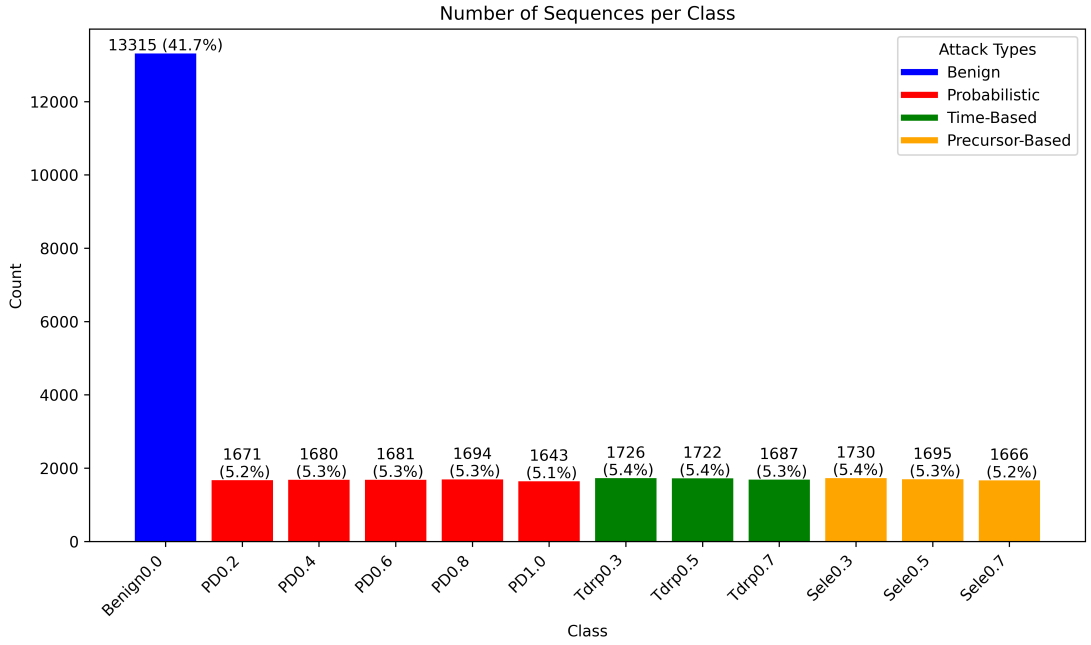


FIGURE 4.2: This bar chart shows the number of sequences per class in FAN-GHETS24.

strategies. The system will automatically run all unique combinations of attack and benign strategies. As is evident, we only have one benign strategy within the defense configuration file, which monitors and logs network information. However, if another monitoring protocol was developed, then this protocol can be specified in the benign configuration file. The same technique could also be used for the attack configuration file, if one wished to write a custom NS-3 AODV protocol which allows another form of attacking behaviour. Next, we use a python script to merge the benign and attack configuration *.ini files and generate a simulation configuration file for every simulation which needs to be run.

Simulation configuration files specify a random seed and the environment parameters such as the maximum node speed and the number of attack nodes within the simulation. The random seed determines the placement, speed and direction of all mobile nodes as the random way point mobility model generates its mobility parameters from this random seed. We supply a template simulation configuration file, "SimulationConfig.ini" which can be modified if needed, however, we provide a python script named "RunSimulations.py" which generates a set of simulation configuration files and automatically runs with with the compiled NS-3 executable. These simulations run in parallel, with each simulation producing two CSV files. The first CSV contains the raw data used for the creation of the dataset, and the second is an attack CSV file, which stores the timestamp of when an attack is executed.

The environment options within this file are limited to the SAR scenario. However, for users wishing to create a custom FANET scenario, we highlight three key functions in the "sim.cc" file where modifications can be made to support this.

Firstly, the `Create_Nodes()` function creates the nodes, malicious and benign node groups, and, assigns the nodes to particular areas of the simulation. This is also where users can customize the mobility model, initial node positions, movement speed, pause time, and the type of random distribution used for positioning. Secondly, the `GenerateTraffic()` function lets you customize the packet sending frequency, quantity, and source/destination sockets. One could modify this function to define a list of destination and origin sockets, allowing the simulation to send to and receive from multiple nodes. Finally, we would like to highlight the `CommandSetup()` function, where one can externalise new environment parameters from the previous two functions, enabling simulation changes without the need for recompilation.

While FAN-GHETS24 provides the necessary sequence data for training ETSC models, it also presents some minor limitations. For instance, the high dimensionality of each record can restrict the number of applicable machine learning techniques which can use this data. For example, while Random Forest classifiers are designed for high dimensionality data, they do not adjust well to time-series data, which makes them inappropriate here. In contrast, deep learning methods appear to be a robust choice of model, as demonstrated in the next section. However, we did find that Long Short-term Memory (LSTM) models struggled with the temporal dimension of FAN-GHETS24. Another limitation of FAN-GHETS24 is the simulation environment. Ideally, the simulation would incorporate more detailed aspects of wireless communication, such as multipath propagation caused by signal reflections from physical objects. However, modelling such phenomena would require a significantly more advanced simulation environment, involving elements like 3D trajectory planning and collision detection for the UAVs. This level of physical modelling falls beyond the scope of our work, as our work focuses on evaluating grey hole attacks from a networking and protocol-level perspective.

For clarification, we would like to point out that FAN-GHETS24 and its source code repository does not contain any malicious code which could endanger a user's computer or UAV network. The attack source code in the repository is specific to NS3 simulations and would have to be modified extensively for it to be a risk to real-world UAVs. Furthermore, the attacks presented in this article have been known for some time and therefore could not be used as part of a zero-day attack. Additionally, the data within FAN-GHETS24 can be summarised as a log of interactions between two UAV entities, making it unsuitable for use as part of an attack.

4.3 Confidence-based Early Classification Model (CEC-FANET)

Henceforth, sequences of node interactions are referred to as *multi-feature time series* (MTSs), representing interactions between the eval node and the subject node. Each

MTS is classified based on the subject node's behavior, which can be one of the 12 specific class labels from Table 5.2. CEC-FANET is trained with both the attack type and intensity as labels. In addition, CEC-FANET operates in three stages:

In Stage 1, an autoencoder is trained on individual time steps from the training dataset's time series. The validation dataset is used to evaluate the performance of this autoencoder. And, the autoencoder serves as a preliminary step for all subsequent base classifiers.

In Stage 2, several convolutional neural network (CNN) models, or base classifiers, are trained to classify sequences of these compact feature representations. Each successive base classifier is responsible for classifying an increasing segment of the time series. For example, the first classifier might operate on the first $\tau = 2$ time steps of an MTS, the next on $\tau = 4$ time steps, the next on $\tau = 6$, and so on. In this way, each classifier processes progressively larger sections of the time series, building upon the earlier segments to improve classification accuracy and confidence.

In Stage 3, the confidence of a classification is determined from both the classification result of an unseen MTS and the *confidence matrix* of the base classifier. The confidence matrix, generated using the training dataset, provides a measure of how reliable a classification is. To determine an appropriate classification threshold, a range of confidence threshold candidates is generated from the validation dataset. These thresholds indicate how confident a model must be to make a classification decision. If the confidence value exceeds the selected threshold, the time series is classified. Otherwise, the decision is deferred until more data is available. The optimal threshold is chosen by balancing classification accuracy with timeliness.

CEC-FANET establishes a suitable benchmark for both Accuracy and Decision Latency, which can be used to evaluate future ETSC models built on this dataset. The remainder of this section provides a detailed description of the model itself. However, before delving into the model details, we first introduce some formal notation.

4.3.1 Early Time-Series Classification Preliminaries

Each MTS consists of a number of time steps, t, where, at each t, a vector of features is present. The maximum defined sequence length, $\mathbb{T}$, denotes the longest period over which an MTS can be classified, though earlier classifications are possible. To formalize this concept, we define an MTS, X, as follows:

Definition 4.1 (Multi-feature Time Series (MTS))**.** An MTS, X, is defined as a matrix $X \in \mathbb{R}^{\tau \times |V|}$, where V is the set of features and τ is the length of the multi-feature time series where $\tau \leq \mathbb{T}$. Standard matrix indexing applies, i.e., $X_{t,:}$ represents the vector of features at time step t, and $X_{:,v}$ represents the time series of feature v, $\forall t \in \{1, 2, \ldots, \tau\}$.

Within the FAN-GHETS24 dataset, MTS lengths may vary: sequences shorter than $\mathbb{T}$ are excluded from this analysis while those exceeding $\mathbb{T}$ are truncated to length $\mathbb{T}$. In the definition of an MTS, τ plays two roles: it represents both the truncated MTS length and the specific *classification point* at which an MTS can be classified with a specific base classifier. We define these points formally below:

Definition 4.2 (Classification Points)**.** A set of classification points, $\mathcal{T} \subseteq \{1, 2, \ldots, \mathbb{T}\}$, represent the specific time steps where the MTS could be classified.

In real-time scenarios, this early classification model would wait for more data to arrive before the next classification is made. However, in FAN-GHETS24, the complete MTS sequence, of length $\mathbb{T}$, is available straight away. Therefore, we emulate the data collection process by masking the time steps from τ to $\mathbb{T}$. Then, if the early classification algorithm requires another classification at the next τ, the next portion of time steps is made available to the model. Like other classification models, this validation model requires the FAN-GHETS24 dataset be divided into three distinct subsets: Training (80%), Validation (10%), and Testing (10%), to provide evidence of model performance. These datasets are formally described below:

Definition 4.3 (Dataset)**.** Given a set of class labels, $\mathcal{Y}$, a collection of MTSs, $\mathbf{X} = \langle X^{(1)}, X^{(2)}, \ldots, X^{(n)} \rangle$, and their associated class labels, $\mathbf{Y} = \langle y^{(1)}, y^{(2)}, \ldots, y^{(n)} \rangle$, a dataset is defined as $\mathbf{D} = \langle \langle X^{(i)}, y^{(i)} \rangle \mid y^{(i)} \in \mathcal{Y}, i = 1, 2, \ldots, n \rangle$, where n is the number of MTSs in the dataset. In this work, there are three datasets: $\mathbf{D}_{\text{train}}$, $\mathbf{D}_{\text{test}}$, $\mathbf{D}_{\text{validation}}$, which represent the training, testing, and validation datasets, respectively.

With these datasets defined, we proceed to train a set of base classifiers, each responsible for making a classification decision at every classification point. Specifically, classifying an MTS at a given point, τ, requires a function that maps the observed sequence up to τ to a class label. This function is formalized below:

Definition 4.4 (Classification of an MTS)**.** Classifying an MTS at classification point τ involves finding a function $\mathcal{H}_\tau : \mathbb{R}^{\tau \times |V|} \to \mathcal{Y}$ that maps the first τ time steps of the MTS to its corresponding class label. An unscripted $\mathcal{H}$ defines the set of all base classifiers across all classification points, i.e., $\mathcal{H} = \{\mathcal{H}_\tau \mid \forall\, \tau \in \mathcal{T}\}$.

To simplify the notation, when classifying an MTS sequence with the classification function, $\mathcal{H}()$, the size of the MTS required is given by the subscript of the classifier. Hence, $\mathcal{H}_\tau(X_{0:\tau,:})$ and $\mathcal{H}_\tau(X)$ are equivalent. We define the early classification of an MTS using a function that identifies both the class label and the specific classification point at which the decision was made:

Definition 4.5 (Early Classification of an MTS)**.** Given a set of base classifiers, $\mathcal{H}$, the early classification of an MTS involves finding a function $\mathcal{F} : \mathbb{R}^{T \times |V|} \to \langle \mathcal{Y}, \mathcal{T} \rangle$. We

define $\mathcal{F}^{\mathcal{Y}}$ as the component of $\mathcal{F}$ that represents the classification output, $\mathcal{Y}$, and $\mathcal{F}^{\mathcal{T}}$ as the component that represents the classification point, $\mathcal{T}$, at which the classification is made.

The early classification model is evaluated with the test dataset using two criteria: Accuracy and Decision Latency, as described below:

Definition 4.6 (Accuracy)**.** Accuracy is the proportion of correctly classified MTSs compared to the total number of MTSs within a dataset, **D**, and is given by:

$$Accuracy(\mathcal{F}) = \frac{||\{i \in \{1,2,\ldots,n\} \mid \mathcal{F}^{\mathcal{Y}}(X^{(i)}) = y^{(i)}\}||}{n}, \tag{4.2}$$

where $\langle X^{(i)}, y^{(i)} \rangle \in \mathbf{D}_{\text{test}}$ and n represents the number of MTSs in $\mathbf{D}_{\text{test}}$.

Definition 4.7 (Decision Latency)**.** Decision Latency is a measure of how early the ETSC model classifies a result. Decision Latency is given by:

$$Decision\_Latency(\mathcal{F}) = \frac{1}{n}\sum_{i=1}^{n}\frac{\mathcal{F}^{\mathcal{T}}(X^{(i)})}{\mathbb{T}}, \tag{4.3}$$

where $X^{(i)} \in \mathbf{D}_{\text{test}}$ and n represents the number of MTSs in $\mathbf{D}_{\text{test}}$.

4.3.2 Dimensionality Reduction

Dimensionality reduction ensures that the base classifiers operate on a low-dimensional representation of the data, making training more efficient. The autoencoder architecture, summarised by Goodfellow et al. (2016), is used here to perform this dimensionality reduction. Specifically, an autoencoder is comprised of two parts, an *encoder*, $\mathcal{E}(.)$, and a *decoder*, $\mathcal{D}(.)$, where the output of the encoder, h, is a latent variable which represents the encoded input, s: $h = \mathcal{E}(s)$. To ensure that h represents a reduced form of s, a decoder function, $\hat{s} = \mathcal{D}(h)$, uses the latent representation as input and attempts to recreate the original input. Training of the autoencoder is then a case of minimising the reconstruction loss, $L_{\text{recon}}(s, \hat{s})$, using standard backpropogation and batch training techniques.

The input features to the autoencoder contain two different types of data, categorical and continuous, therefore, we found it pertinent to introduce a loss function for each type. Specifically, we introduce a categorical loss function, L_{cat}, and a continuous loss function, L_{con}, making the reconstruction loss the total of both losses:

$$L_{\text{recon}} = L_{\text{cat}} + L_{\text{con}}, \tag{4.4}$$

where L_{con} is the mean squared error loss function and the L_{cat} is the BCELossLogits loss function.

Input Dim	Output Dim	Description
Categorical Branch		
input_cat_dim	64	Linear + BatchNorm + ReLU
64	32	Linear + BatchNorm + ReLU
Continuous Branch		
input_con_dim	64	Linear + BatchNorm + ReLU
64	32	Linear + BatchNorm + ReLU
Combined Encoder		
64	30	Linear + BatchNorm + ReLU
30	12	Linear + BatchNorm + ReLU
12	6	Linear + BatchNorm + Sigmoid
Combined Decoder		
6	12	Linear + BatchNorm + ReLU
12	30	Linear + BatchNorm + ReLU
Categorical Decoder Head		
30	64	Linear + BatchNorm + ReLU
64	input_cat_dim	Linear
Continuous Decoder Head		
30	64	Linear + BatchNorm + ReLU
64	input_con_dim	Linear + BatchNorm + Sigmoid

TABLE 4.6: Autoencoder Architecture

The autoencoder architecture used in this work is described in Table 4.6 and comprises a multi-branch encoder design which allows different feature types to be processed through the network separately, before being concatenated and reduced to the embedding dimension. Likewise, the decoder has a multi-headed design with different activation functions for the reconstructed feature types.

The autoencoder is applied to every time step of the MTS, i.e., an MTS sequence of size $\mathbb{R}^{\tau \times |V|}$ becomes an *encoded* MTS of size $\mathbb{R}^{\tau \times |h|}$.

4.3.3 CNN Classifier

Encoded MTSs are classified using a convolutional neural network. We train $|\mathcal{T}|$ base classifiers using $\mathbf{D}_{\text{train}}$. Each classification point, $\tau \in \mathcal{T}$, has an associated classifier, $\mathcal{H}_\tau$, which is comprised of the autoencoder from the previous section and a CNN classifier.

We experimented with a range of representations. However, the most successful representation involved maintaining the time dimension and treating the $|h|$ features as input channels. We then use 1D convolutional layers to identify patterns in the channels

Input Channels	Output Channels	Kernel Size	Padding	Stride	Description
6	384	65	32	32	Conv1D, BatchNorm1D, LeakyReLU
384	768	33	16	16	Conv1D, BatchNorm1D, LeakyReLU
768	1536	5	2	2	Conv1D, BatchNorm1D, LeakyReLU
1536	1536	5	2	2	Conv1D, BatchNorm1D, LeakyReLU
-	-	-	-	-	Flatten
12288	3072	-	-	-	FC* 1, BatchNorm1D, LeakyReLU
3072	1536	-	-	-	FC* 2, BatchNorm1D, LeakyReLU
1536	12	-	-	-	FC* 3

*Fully Connected Layer

TABLE 4.7: CNNClassifier Architecture

which could be associated with a particular class. Table 4.7 summarises the classifier architecture used for all base classifiers.

4.3.4 Classification Confidence

We adopt an approach first proposed by Lv et al. (2019), which uses a *confidence* value derived from a confidence function, $C()$, and a *confidence threshold*, $\phi*$, to evaluate a classification prediction at each τ. If the confidence value of a prediction does not meet the confidence threshold, then the next τ in sequence is chosen and another classification is made until the end of the sequence is reached or the confidence level meets or exceeds the threshold. Interestingly, as well as training the base classifiers with $\mathbf{D}_{\text{train}}$ we can also use the same training dataset to generate a confusion matrix. In traditional classification settings, the test dataset is used to generate this confusion matrix to demonstrate the performance of the classifier on unseen data. However, using a confusion matrix derived from the training dataset allows us to quantify the model's certainty in its predictions, providing an insight into the reliability of classification outcomes. To differentiate these concepts, we will call a *confidence matrix* a matrix of probability values derived from the confusion matrix of the training dataset.

As an example, say we wanted to generate a confidence matrix by using a trained classifier in a two class classification problem ($\mathcal{Y} = \{y_1, y_2\}$). To start, the model predicts $\hat{y} = y_1$ for an unseen data point. By referencing values in the confusion matrix, we can start to appreciate how accurate that result is. Moreover, we can generate probabilities from these values if we divide by how many predictions of that label were

made using the classifier. This table of probabilities is our confidence matrix. Continuing our example, let us say there are 4 instances in our hypothetical training dataset where the classifier predicted y_1: 3 of them are correct and the other is incorrect. Ergo, if the classifier predicts y_1, the probability of a correct classification, $P(y_1|\hat{y} = y_1)$, is 75%, while the probability of an incorrect classification is $P(y_2|\hat{y} = y_1)$, is 25%. Expanding this example to the general case, we arrive at the *performance* of a classifier, which is defined as:

$$\pi_{\mathcal{H}_\tau}(y|\hat{y}) = \frac{||\{i \in \{1,2,\ldots,n\} \mid \mathcal{H}_\tau(X^{(i)}) = \hat{y} \;\&\; y^{(i)} = y\}||}{||\{i \in \{1,2,\ldots,n\} \mid \mathcal{H}_\tau(X^{(i)}) = \hat{y}\}||}, \tag{4.5}$$

where n is the number of MTS sequences in the training dataset. Equation 4.5 calculates the probability that y is the true class label given that $\hat{y}$ is predicted for a particular base classifier. In other words, this equation is a formal representation of how we use the confidence matrix to generate the probabilities associated with a specific classification. The denominator of Equation 4.5 is the set of indices which represent all of the instances within the training dataset where the classifier has predicted $\hat{y}$, i.e. the sum of the $\hat{y}$ column within the confusion matrix. Similarly, the numerator of Equation 4.5 represents the set of indices where $\hat{y}$ was predicted by the classifier and the actual class label was y. While calculating the confidence value, the confidence matrix can be thought of as a lookup table to calculate the classifier performance when a new prediction, $\hat{y}$, is made using an unseen MTS. In essence, for any given y, $\pi_{\mathcal{H}}(y \mid \hat{y})$ measures the probability that the classifier is wrong if $y \neq \hat{y}$ or the probability that the classifier is correct if $y = \hat{y}$.

For a single prediction at τ, the performance of a classification result can be used as a confidence value by using the following equation:

$$C_\tau(\hat{y}) = \pi_{\mathcal{H}_\tau}(\hat{y} \mid \hat{y}). \tag{4.6}$$

However, more appropriately, a single MTS can be inferred through successive base classifiers at certain classification points, i.e. $\mathcal{H}_{\tau_1}, \mathcal{H}_{\tau_2}, \mathcal{H}_{\tau_3}, \ldots, \mathcal{H}_{\tau_T}$. Therefore, a way of amalgamating multiple confidence values together is needed to consider previous classifications of the MTS. This can be done with the following equation:

$$C_\tau(\mathcal{H}_\tau(X)) = 1 - \prod_{\tau' \in \mathcal{T},\, \tau' \leq \tau} \left(1 - \pi_{\mathcal{H}_{\tau'}}(\mathcal{H}_\tau(X) \mid \mathcal{H}_{\tau'}(X))\right). \tag{4.7}$$

To explain further, Equation 4.7 calculates the confidence of multiple classification results at certain classification points. Note that the Xs in this equation can be from $\mathbf{D}_{\text{validation}}$ or $\mathbf{D}_{\text{test}}$ but not $\mathbf{D}_{\text{train}}$. As an example of how Equation 4.7 is used, let us assume that there are two classification points at time steps 5 and 10, $\mathcal{T} = \{5, 10\}$. With these values in mind, unrolling Equation 4.7 gives:

$$C_{10}(\mathcal{H}_{10}(X)) = 1 - [(1 - \pi_{\mathcal{H}_5}(\mathcal{H}_{10}(X)|\mathcal{H}_5(X))) \times (1 - \pi_{\mathcal{H}_{10}}(\mathcal{H}_{10}(X)|\mathcal{H}_{10}(X)))], \tag{4.8}$$

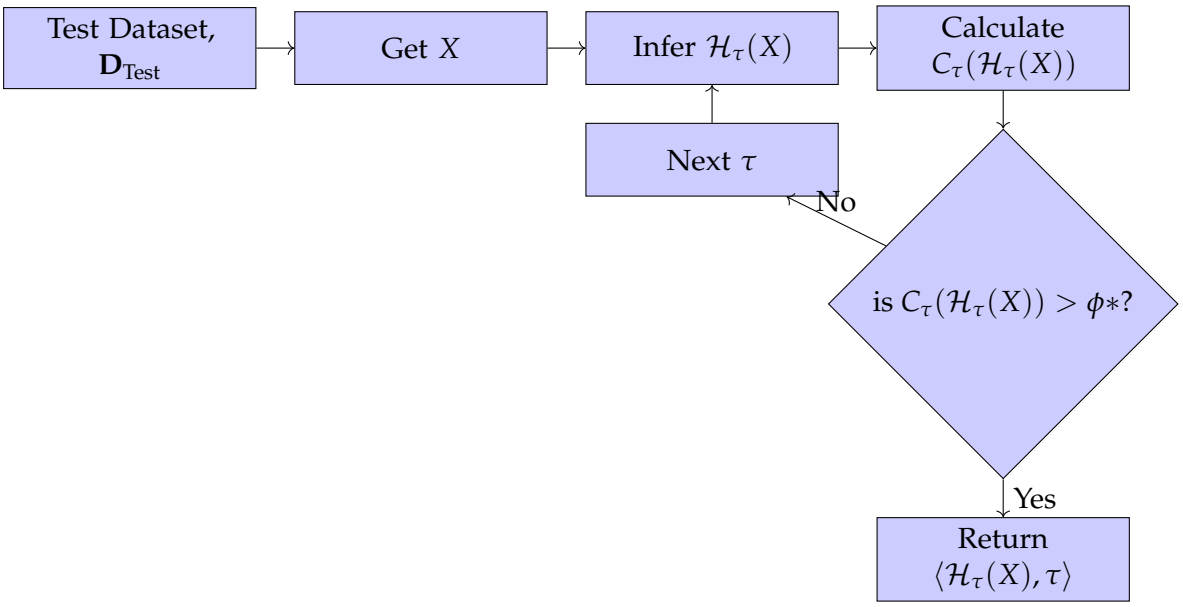


FIGURE 4.3: This figure shows the operation of the early classification function ($\mathcal{F}$) and how the base classifiers interact with the confidence threshold. This figure has been adapted from a figure by Lv et al. (2019).

which evaluates the confidence of a prediction result, $\mathcal{H}_{10}(X)$, given that $\mathcal{H}_5(X)$ is predicted in a previous time step.

The early classifier, $\mathcal{F}$, uses successive base classifiers as well as a confidence threshold, $\phi*$, to indicate when classification should stop and return the final $\langle \hat{y}, \tau \rangle$ pair. In live operation, the early classifier will start by waiting τ time steps to collect incoming data. The resulting τ time steps are classified using the appropriate classifier and the confidence is calculated. If the confidence level exceeds $\phi*$, the final classification result and current time step is returned. If the confidence level does not meet this threshold, more time steps are requested and another classification is made at the next τ in sequence. This process repeats until a classification and time step is returned. If the confidence never exceeds the confidence threshold, then the result of the last base classifier is used as a classification result. This algorithm is described by Figure 4.3.

It is clear that the choice of $\phi*$ makes a considerable difference to the Accuracy and Decision Latency of the early classifier. Therefore, a set of candidates, $\phi \in \Phi$, is generated by calculating all the confidence values for every MTS at every $\tau \in \mathcal{T}$ in the validation dataset. Duplicate confidence values are removed from the list and the remaining values are sorted in ascending order. The candidate set is then formed by taking the mean of every two consecutive values from the sorted confidence value list. This allows us to create midpoints between known confidence values to use as candidate thresholds.

Model	Hyperparameter	Value
Autoencoder	Learning Rate	1e-3
	Number of Epochs	45
	Batch Size	512
CNN Classifier	Learning Rate	1e-5
	Number of Epochs	120
	Weight Decay	1e-3
	Batch Size	64

TABLE 4.8: Autoencoder and Base Classifier Hyperparameters

Each $\phi \in \Phi$ is tested in the *confidence loss function*:

$$L_{\text{conf}}(\phi) = \kappa(1 - Accuracy(\mathcal{F})) + (1 - \kappa)Decision\_Latency(\mathcal{F}), \tag{4.9}$$

where κ is a hyper parameter used to define the trade off between Accuracy and Decision Latency with the validation dataset used for the calculation of Accuracy and Decision Latency.

An optimal $\phi*$ is chosen which minimises the confidence loss function:

$$\phi* = \underset{\phi \in \Phi}{\text{argmin}} \left[L_{\text{conf}}(\phi) \right]. \tag{4.10}$$

4.4 Experimental Evaluation

To evaluate the validation model, let $\mathcal{T} = \{1000, 2000, 3000, \ldots, 16000\}$). The test dataset, $\mathbf{D}_{\text{test}}$, is used in this section to ensure that none of the training or validation MTSs are used for the experimental evaluation. The ETSC studies by Lv et al. (2019); Mori et al. (2018) report that a κ value of 0.9 leads to the best model performance. This was verified by our own experiments, as seen in Table 4.9. For the autoencoder, we define $|h| = 6$ as the reconstruction loss during training was observed to be minimal. We are cognizant of the fact that these values can be tuned more accurately for increased model performance. However, CEC-FANET showcases the dataset's applicability, rather than emphasize any particular performance results. The hyperparameters for the autoencoder and all base classifiers are detailed in Table 4.8.

Using the parameters defined in Section 4.4, we present the results of each stage of CEC-FANET. We start with the dimensionality reduction component in Section 4.4.1 and move on to describe the results of the base classifiers in Section 4.4.2. Although 16 base classifiers were trained, we only show the results of two base classifiers. These particular classifiers are necessary to compare and contrast the performance of the early classifier in Section 4.4.3.

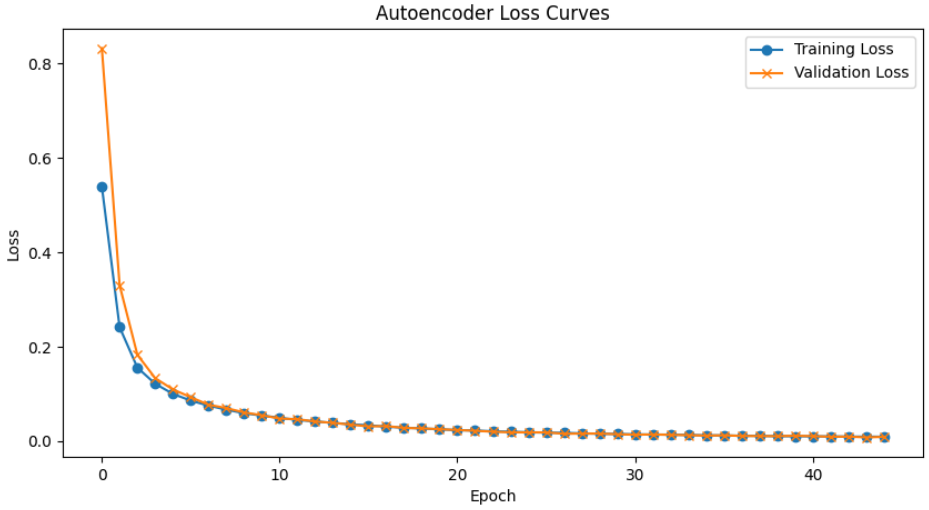


FIGURE 4.4: This graph shows the learning curve of the autoencoder.

4.4.1 Autoencoder Training

Figure 4.4 shows the training curve of the autoencoder. This training curve demonstrates that the autoencoder is able to recreate high dimensional data from a low dimensional embedding space.

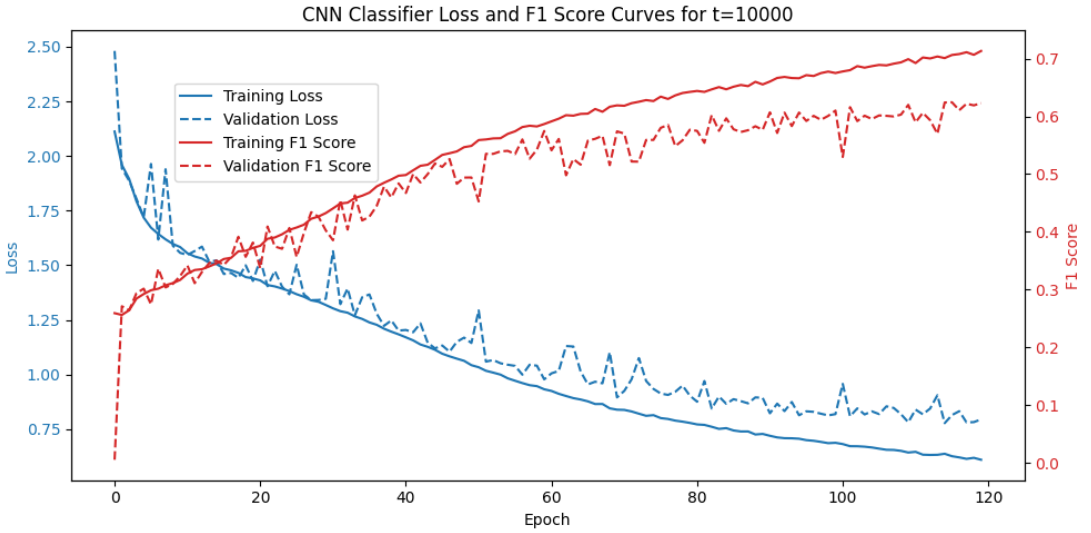
4.4.2 Base Classifier Training

Figure 4.5 shows the training graphs of 2 base classifiers, demonstrating how both F1 score increases and loss decreases with increasing epochs in all cases. These learning curves provide evidence that FAN-GHETS24 includes features which enable the classification of different types and intensities of gray hole attacks. The F1 score is widely used for evaluating classification models, however, Accuracy is more commonly employed for ETSC models. Thus, we present the performance of the base classifier using the F1 score here, but subsequent results will be measured using Accuracy.

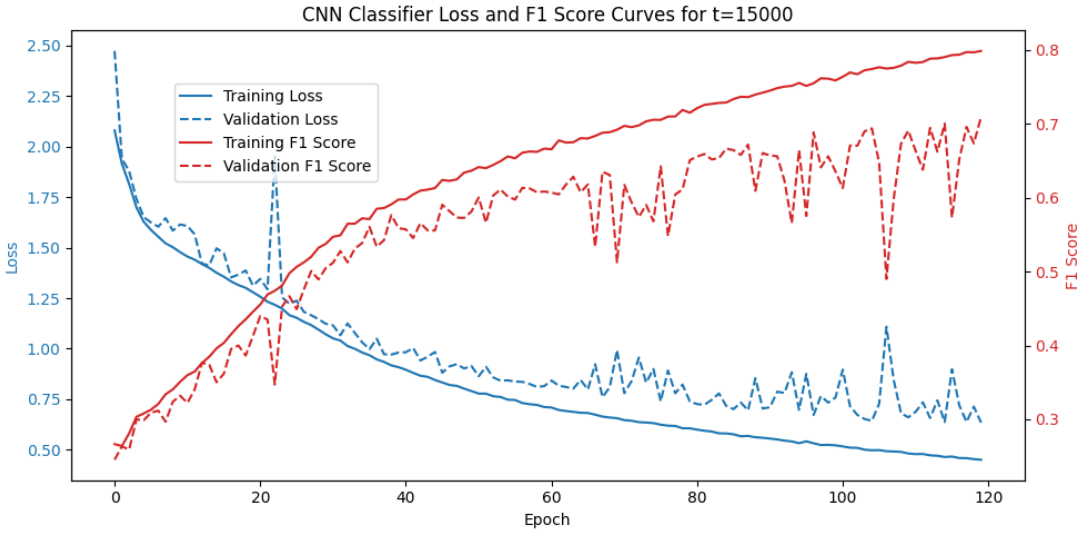
Figure 4.6 shows how Accuracy varies with τ for all base classifiers using the validation dataset. As expected, the longer the MTS segment, the better the accuracy of the classifier. In addition, Figure 4.7 shows the confusion matrix of classifier $\mathcal{H}_{t=15000}$ using the validation dataset.

4.4.3 Proposed Model

Table 4.9 shows the Accuracy and Decision Latency results of our ETSC model when κ = 0.7, 0.8 and 0.9. We also show the results of 2 base classifiers, where $\tau = 10000$ and



(A) $\tau = 10000$



(B) $\tau = 15000$

FIGURE 4.5: These graphs show the training progress of two base classifiers: $\mathcal{H}_{\tau=10000}$ and $\mathcal{H}_{\tau=15000}$. Each graph indicates the F1 score and loss for both the training and validation partition of the FAN-GHETS24 dataset.

$\tau = 15000$, and compare these results with our ETSC models. We can conclude that $\kappa = 0.9$ produces the model with the best Decision Latency and Accuracy score. Consequently, we can compare this result with our base classifiers to justify the utility of ETSC models. The Decision Latency values for the ETSC models are generated by Equation 4.3, which calculates the average classification point when using the ETSC model to classify all MTSs in $\mathbf{D}_{\text{test}}$. In contrast, the Decision Latency of the base classifiers is fixed for every MTS, as there are a fixed number of time steps which the base classifier acts on.

Comparing the ETSC results with a base classifier of similar Decision Latency, $\mathcal{H}_{\tau=10000}$,

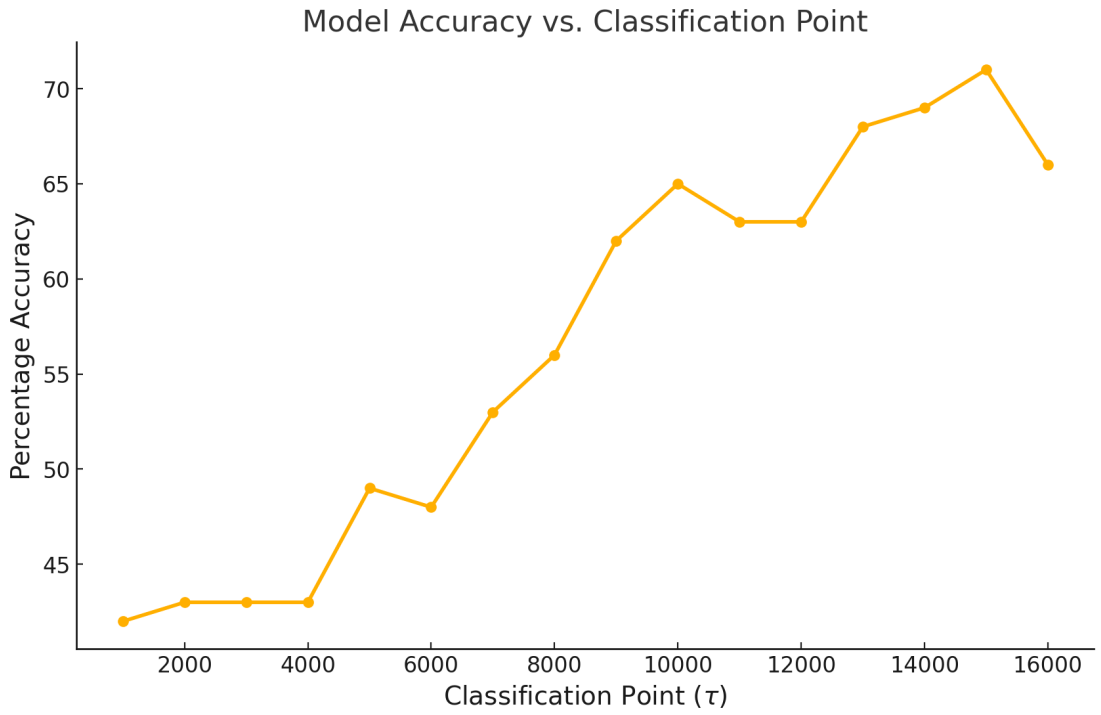


FIGURE 4.6: This figure illustrates the relationship between the Accuracy and the length of the MTS segments which are used to train the base classifiers. As the segment length increases, the Accuracy score generally improves, indicating that longer training segments enhance the classifiers' performance, although the trend exhibits some variability at higher sequence lengths.

the early classifier demonstrates an Accuracy improvement of 5%. In addition, while the Accuracy of the other classifier, $\mathcal{H}_{\tau=15000}$, is 1% higher, the Decision Latency increases by 32%. This result indicates that while the ETSC model is 1% less Accurate, we can classify sequences much faster than the base classifier. In summary, these results indicate that FAN-GHETS24 is a good candidate for ETSC, as any fixed time sequence classification methods produce notably worse results than the ETSC equivalent.

Figure 4.8 shows an example of a MTS that was correctly classified as **PD1.0** by CEC-FANET. Owing to the intensity and clear signature of the GHA, the model was able to make a confident prediction halfway through the sequence, before the stated average classification point typically observed (62%).

Usage Notes

One of the main challenges in using this dataset is the quantity of data captured by each eval node. Much of this data is redundant, therefore, a way of intelligently filtering this data before classification may be beneficial. Alternatively, due to the number of benign MTSs, this dataset could also be treated as an anomaly detection problem before finally

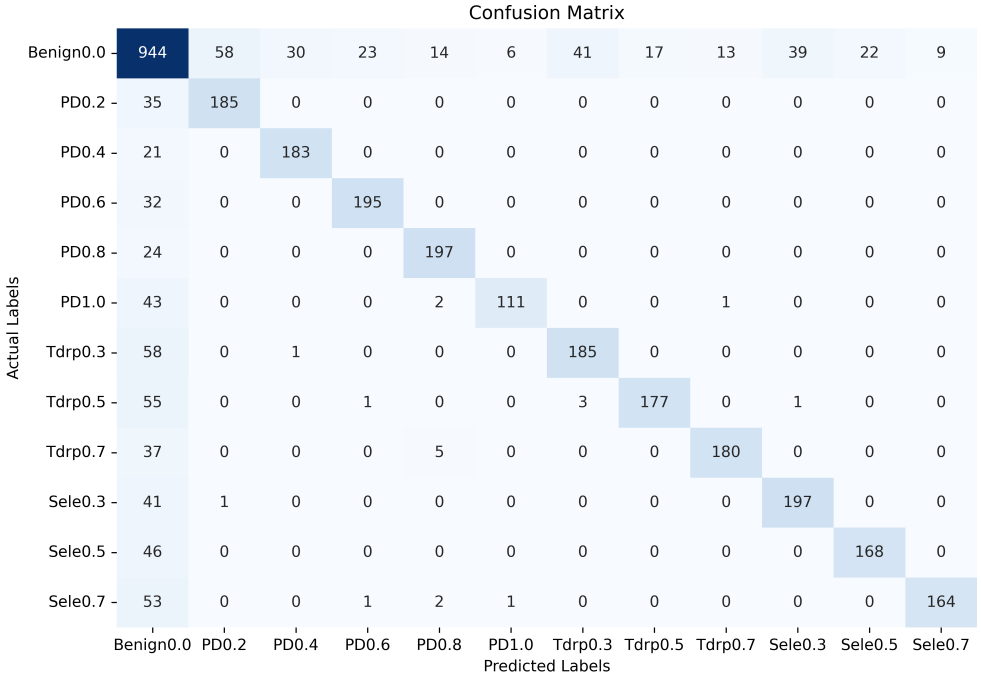


FIGURE 4.7: This figure shows the confusion matrix of classifier $\mathcal{H}_{t=15000}$

Classifier	**Accuracy**	**Decision Latency**
Base Classifier ($\mathcal{H}_{\tau=10000}$)	65%	63%
Base Classifier ($\mathcal{H}_{\tau=15000}$)	71%	94%
Early Classifier ($\mathcal{F}$, $\kappa = 0.9$)	**70%**	**62%**
Early Classifier ($\mathcal{F}$, $\kappa = 0.8$)	69%	62%
Early Classifier ($\mathcal{F}$, $\kappa = 0.7$)	68%	62%

TABLE 4.9: This table shows the Accuracy and Decision Latency of 3 early classifiers each implemented with three different κ values, $\kappa = 0.9$, $\kappa = 0.8$ and $\kappa = 0.7$. These early classifiers are compared with the results of 2 base classifiers, $\mathcal{H}_{\tau=10000}$ and $\mathcal{H}_{\tau=15000}$. A high Accuracy with a low Decision Latency indicates a better model.

classifying the anomaly type (anomalies in this case would refer to types and intensities of grey hole attacks). Our repository:

https://git.soton.ac.uk/ch1u20/fan-ghets24/,

contains the source code for generating FAN-GHETS24. This repository allows researchers to regenerate the dataset and explore different threat models by creating additional NS3 protocols. Alternatively, if one wishes to download the dataset as it stands, FAN-GHETS24 is publicly available at this link:

https://doi.org/10.5281/zenodo.13315419,

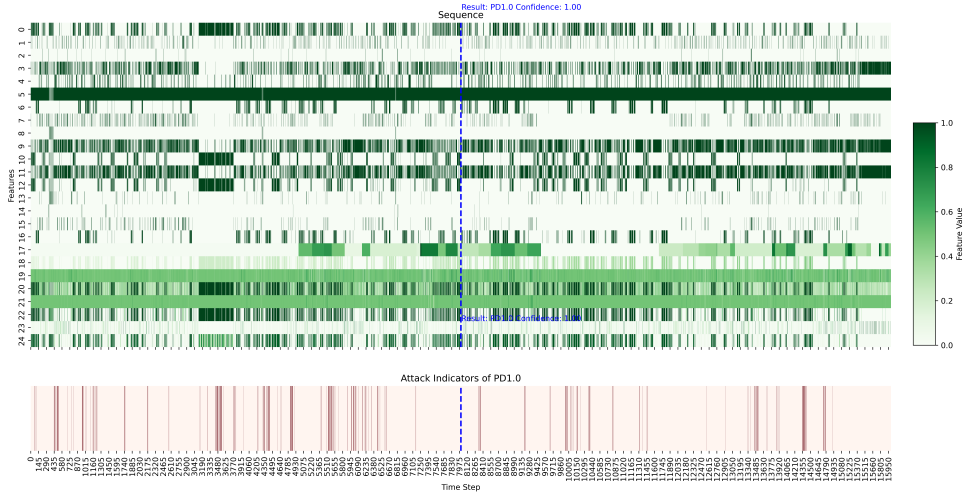


FIGURE 4.8: This figure shows a successful classification of **PD-1.0** at $\tau = 8000$ (Blue dotted line), representing an Decision Latency of 50%. The upper area of the graph represents the value of each feature, while the lower portion of the graph presents the attack indicators. The attack indicators show when a packet has been dropped maliciously.

We note the class imbalance of FAN-GHETS24 and actively recommend not to balance the classes, as was the case in CEC-FANET. This is because the statistical distribution of real-world occurrences must be faithfully represented in the training dataset, as stated by Weiss and Provost (2003). Artificially altering the class distribution risks distorting the model's predictive capabilities, leading to reduced accuracy.

Ultimately, we recommend using every time step from each sequence when training models with the FAN-GHETS24 dataset. However, similar to previous methods, users may choose to average the metrics over a fixed time window, resulting in a smaller dataset which can be used to train smaller models. If this approach is taken, each resulting sub-sequence may contain a variable number of attack points. To address this, users may wish to define the minimum number of attack points required for a sub-sequence to be classified as a specific intensity or type of GHA.

In general, we recommend the following process to begin working with this dataset:

1. Train a dimensionality reduction model on each data point within the sequence. There are 25 features in total and, in our experience, we found that classification models work best when using an embedding space.

2. Take note of the different types of data in each sequence. We generated a JSON metadata file which contains this data for each sequence. We recommend using this to ascertain how best to manipulate these variables.

3. Use the Pyarrow Python module in combination with Pandas to read the parquet files. In our experiments, it showed noticeable speed increases. Furthermore, we

also found it useful to use high memory compute nodes during training to store the full dataset and offload this data to the GPUs as each batch is trained.

4. Train a sequence based classification model. In this work, we used a CNN to validate the dataset, however, we are confident that other model types and architectures can be used. We are very interested to see how other researchers utilise this dataset and the models which are produced as a result.

5. Use an early classification system to determine the optimum classification Accuracy and Decision Latency from a sequence of classification predictions. We demonstrated the use of a confidence thresholding based system, however, other systems can be used. For example, Sharma and Kumar Singh (2021) use the class probabilities from the classifier to determine a set of early stopping rules. We hypothesise that future novel systems could be utilised in combination with our dataset to further improve early classification algorithms in FANETs.

4.5 Conclusion

This chapter motivates and describes FAN-GHETS24, a FANET-specific dataset developed to support the training of early time series classification (ETSC) models. This dataset captures the temporal dynamics of packet exchange while accounting for node mobility, and includes a range of GHA types and intensities as class labels. Using this dataset, we train an ETSC model, CEC-FANET, designed to classify these attack patterns based on progressively longer segments of observed time series data. Results demonstrate that the features within FAN-GHETS24 are sufficiently expressive for accurate classification of GHA behaviours. Furthermore, we show that standard base classifiers perform worse than CEC-FANET, highlighting the advantages of ETSC models in this context.

Chapter 5

A Game Theoretic Evaluation Framework for FANETs Against Grey Hole Attacks

While the classification model in Chapter 4 demonstrates that grey hole attacks can be detected from temporal patterns of node interactions, detection alone does not guarantee improved quality of service (QoS) in flying ad hoc networks (FANETs). Effective defence requires coupling detection with appropriate mitigation strategies embedded in the routing protocol. Assessing such strategies is not straightforward, as existing evaluation methods often overlook configuration choices, trade-offs across different attack types, and variability in network conditions. To address this, this chapter introduces FANET-Rank, a game-theoretic evaluation framework that provides a structured and comprehensive means of comparing defence protocols under diverse attack and mobility scenarios.

5.1 Introduction

It is crucial to mitigate grey hole attacks and assess the effectiveness of these mitigation efforts to ensure that FANETs can carry out their mission. As mentioned in Section 2.9, various techniques have been employed within defence protocols to mitigate grey hole attacks in FANETs. And, in these works, the proposed protocols are evaluated to ensure state of the art performance. In a typical evaluation procedure, the proposed protocol and several baseline defence protocols are simulated under identical attack conditions. Several environments are generated to test the proposed protocol's resilience and each simulation is executed multiple times using different seeds. Once simulations are complete, the performance metrics are gathered and averaged per protocol. The

expectation is that the proposed protocol will demonstrate superior performance when compared with the baseline defence protocols.

5.1.1 Motivation

This evaluation procedure appears reasonable and robust; however, there are **four** overlooked issues which substantially limit the scope of the evaluation and can lead to incorrect conclusions regarding which protocol performs the best. These issues are described in more detail in Sections 5.1.1.1, 5.1.1.2, 5.1.1.3 and 5.1.1.4.

5.1.1.1 Unexplored Configuration Space

A configuration parameter's influence could have a dramatic effect on the output of the protocol in some scenarios and therefore, it would be pertinent to test a range of configuration parameters. Defence configuration parameters are typically chosen based on expert knowledge or empirical evidence (e.g. a trust threshold), whereas attack configuration parameters are commonly introduced as part of the threat model (e.g. % packet drop rates).

prior studies exclusively present their findings based on a single set of configuration parameters (Mukherjee et al., 2018; Sánchez-Casado et al., 2015; Wei et al., 2014; Bounouni et al., 2022; Singh et al., 2020). For example, a model proposed by Singh et al. (2020) defines the membership functions based on Gaussian curves, which affects how their packet forwarding ratio influences their final decision. However, the authors do not study the effects of implementing alternative membership functions, such as trapezoidal shaped functions which could the reduce the computational load without a reduction in performance. An investigation into the performance difference between membership functions could be practically implemented by modifying a configuration parameter. Likewise, other studies also fix important values, such as the threshold for classifying nodes as malicious or benign (Sánchez-Casado et al., 2015; Tropea et al., 2024) or decay rates that control how node reputations change over time (Bounouni et al., 2022). We stipulate that investigating the effects of these configurable parameters is essential, as incorporating them into the evaluation methodology provides a more comprehensive depiction of the protocol's capabilities. In previous works, configuration parameters are often chosen empirically, with little or no justification provided. Including these parameters explicitly within the evaluation process offers greater transparency and insight.

5.1.1.2 Performance Variability under Different Threats

In addition, some works fail to state the intensity of their grey hole attack (Wei et al., 2014; Singh et al., 2020; Gurung and Chauhan, 2018; Chandrasekar et al., 2024), whilst others only use a single intensity value (Jari et al., 2021; Rani et al., 2020; Ullah and Das, 2018). Some works evaluate their protocols using a limited range of intensities (Bounouni et al., 2022; Sadayan and Ramaiah, 2022; Tropea et al., 2024), but their evaluation procedures do not consider cases where their protocol's performance may degrade at certain intensity levels. In their studies, this issue fortunately did not arise.

An analysis of different intensity values is significant because, due to the operation of some defence protocols, it may be advantageous for the attacker to choose a grey hole attack that maximizes damage over the FANET's lifetime. For example, attacks which employ a lower intensity may, counter-intuitively, have a greater impact on the QoS of the network compared with an attack of greater intensity. For instance, a study by Bounouni et al. (2022) states that a node employing a packet drop rate of 50% can potentially drop a larger quantity of packets compared with a packet drop rate of 75%. This is because some defensive protocols are more capable of defending against more obvious attacks, making the identification and rejection of malicious nodes easier. Subsequently, over the course of a simulation, more packets are transferred successfully, leading to an unexpectedly high packet delivery rate for environments which have high intensity malicious nodes within them. Because of these scenarios, it is important to evaluate grey hole attacks across different intensities; however, this aspect remains relatively unexplored in current evaluation methods. More worryingly, none of the literature we reviewed considered time-based or precursor-based grey hole attacks. Probabilistic grey hole attacks remain the most tested attack in literature, meaning that a large part of the grey hole threat landscape is ignored.

5.1.1.3 Performance Uncertainty

Furthermore, we did not observe any works which took into consideration the performance variance due to node placement and speed. In most studies we reviewed, repeated experiments were not performed (Singh et al., 2020; Sadayan and Ramaiah, 2022; Wei et al., 2014; Bounouni et al., 2022; Chandrasekar et al., 2024). While only one related work (Tropea et al., 2024) repeated their experiments 15 times and averaged the result. According to Tom Henderson et al. (2008), creator of NS-3, it is highly recommend to conduct multiple simulation runs with different seeds.

However, relying solely on mean performance over multiple seeded runs fails to account for the variability caused by randomized node placements and velocities. A more informative performance metric may include both the mean and the variance, capturing not only the expected behaviour but also the consistency of that behaviour

across different simulation conditions. Ultimately, the most meaningful indicator of reliable performance is not just a high average, but the highest level of performance that can be maintained consistently.

5.1.1.4 Unexplored Environment Space

Typical protocol evaluation involves varying one environmental factor while holding others constant (Singh et al., 2020; Bounouni et al., 2022; Jinarajadasa et al., 2018; Ullah and Das, 2018; Chandrasekar et al., 2024; Wei et al., 2014; Sadayan and Ramaiah, 2022; Jari et al., 2021). This approach leaves large regions of the environmental parameter space unexplored, which may obscure behaviours or weaknesses that only arise under specific combinations of conditions.

Varying environment parameters is essential to demonstrate the defence protocol's effectiveness in a range of different scenarios. However, works in this field only investigate certain sections of this environment parameter space. For example, some works (Singh et al., 2020; Bounouni et al., 2022; Jinarajadasa et al., 2018; Ullah and Das, 2018; Chandrasekar et al., 2024) give a detailed explanation of their protocols' effectiveness in environments with different ANRs, but fail to mention how maximum node speed influences the operation of their respective protocols. On the other hand, other works (Wei et al., 2014; Sadayan and Ramaiah, 2022; Jari et al., 2021; Singh and Verma, 2020) evaluate their protocol based on the maximum node speed and the malicious node ratio. However, one parameter is fixed while the other is varied.

5.1.2 Contributions

To address these 4 shortfalls, this work proposes the FANET-Rank evaluation framework, which implements *Empirical Game Theoretic Analysis* (EGTA) (Wellman and Prakash, 2014), where performance metrics from simulations can be used with game theoretic methods to quantify the effectiveness of defence protocols, and their configuration parameters, across the entire threat landscape. We use a metric called the *regret* (specifically the max regret), which allows us to measure the performance drop experienced by a defence protocol relative to other defence protocols within the evaluation. Using regret provides a suitable evaluation metric which balances raw performance while taking into consideration weaknesses in certain areas of the threat landscape.

To account for the variance in performance with different node placements and velocities, we introduce bootstrapping (Davison and Hinkley, 1997). By resampling from the available simulation results, bootstrapping generates more empirical games for EGTA, allowing a more precise value of regret. Combining bootstrapping and EGTA in

this way allows one to evaluate how node mobility and protocol behaviour affects QoS under adversarial conditions.

In addition, we extend protocol evaluation by jointly varying both the maximum node speed and the ANR, rather than isolating one while fixing the other. By applying EGTA and bootstrapping across this broader parameter space, we provide a more comprehensive and robust assessment of protocol performance, capturing behaviours that may only emerge under specific combinations of environmental parameters.

To assess FANET-Rank, we implement three defence protocols from previous literature and define a range of attack and defence configuration parameters. These defence protocols are evaluated using both FANET-Rank and a typical evaluation method. We deliberately select a valid, yet least informative, typical evaluation procedure from previous work, in order to highlight the stark contrast between FANET-Rank and conventional evaluation approaches. Specifically, the typical evaluation procedure used in this work orders the defence protocols according to network performance from highest to lowest. Moreover, the protocols are also subjected to one particular grey hole attack in an environment where the number of attack nodes is varied. This evaluation procedure is deemed acceptable within the FANET community. We compare the rankings produced by FANET-Rank and the typical evaluation method to demonstrate how misleading typical evaluations can be. In our first test case, FANET-Rank correctly demotes a particular protocol that causes a 12% decrease in packet delivery rate. This performance decrease would have been ignored under the typical evaluation method. In our second test case, we observe a 10% increase in packet delivery rate by including configuration parameters within the evaluation. In our third test case, we observe changes in regret values of up to 5% due to bootstrapping, which subsequently impact the rankings of the defence protocols. Lastly, in our 4th test case, we show how varying both ANR and maximum node speed influences the defence protocol. This variation would not have been identified by using a typical evaluation method.

5.1.3 Summary

Table 5.1 provides a summary of the procedures used to evaluate defence protocols which mitigate grey hole attacks. In addition, Table 5.1 highlights the areas of improvement addressed by FANET-Rank.

TABLE 5.1: Comparison of Evaluation Methods

Source	Evaluation Environment	Analyses Performance Based on a Range of Different Grey Hole Attack Types and Intensities	Evaluates a Range of Defence Configuration Parameters	Evaluates Both Attack Node Ratio and Maximum Node Speed	Considers the Variability in Node Placement and Velocity
Singh et al. (2020)	Velocity: ?, Num Malicious Nodes: 0-5 Total Nodes: 50	✗	✗	Evaluated Separately	?
Sadayan and Ramaiah (2022)	Velocity: 10m/s fixed, Num Malicious Nodes: 0-10 Total Nodes: 30	✗	✗	Evaluated Separately	?
Wei et al. (2014)	Velocity: 10m/s fixed, Num Malicious Nodes: 2-10 Total Nodes: 5-30	✗	✗	Evaluated Separately	?
Bounouni et al. (2022)	Velocity: 10m/s fixed, Num Malicious Nodes: 2-10 Total Nodes: 40	Intensity Only	✗	Evaluated Separately	?
Tropea et al. (2024)	Velocity: ?, Num Malicious Nodes: 5-25 Total Nodes: 50	Intensity Only	✗	Attack Node Ratio Only	Average Only
Chandrasekar et al. (2024)	Velocity: 20m/s fixed, Num Malicious Nodes: ? Total Nodes: 5-25	✗	✗	Evaluated Separately	?
This Work	Velocity: 5-35m/s, Num Malicious Nodes: 3-9 Total Nodes: 24	✓	✓	✓	Average and Variance

5.2 Methodology

In this Section, we will describe FANET-Rank, a novel evaluation framework for defence protocols. To begin the explanation of the framework, Section 5.2.1 describes the process of constructing an empirical game and the use of the regret metric while Section 5.2.2 describes how bootstrapped regret distributions are used to form the defence strategy rankings. Section 5.2.3 presents an overview of FANET-Rank, accompanied by a step-by-step procedure for its use. Finally, Section 5.2.4 examines the practical aspects of FANET-Rank, including guidance on when and where it is best applied.

5.2.1 Empirical Game Theoretic Analysis (EGTA)

While the game-theoretic methods described in Section 2.9.2 are often used to inform the decision-making of individual nodes, they can also be repurposed as a powerful evaluation framework. Instead of using game theory to dictate how a node should behave, we can model the entire FANET as a competitive interaction between a group of attacker nodes and a group of defender nodes, where each player's strategy represents a combination of protocol and configuration parameter, enabling a broader and more insightful evaluation of the protocols under test.

Traditional evaluation methods typically assess performance metrics under fixed attacker models, which does not reveal whether a protocol is fundamentally fragile against other threats. Even when performance is high in one case, a protocol that collapses under another type of attack should not be ranked highly. This is why we must balance outright performance with resilience against all threats. By viewing evaluation as a strategic contest, game theory allows us to reason about vulnerability in worst-case outcomes. Moreover, Empirical Game-Theoretic Analysis (EGTA) (Wellman, 2006; Wellman et al., 2025) provides a principled method for conducting this kind of evaluation. EGTA allows us to take simulation data, construct empirical payoff tables and analyse the games using traditional game theoretic techniques, similar to previous works (Prakash and Wellman, 2015; Wellman et al., 2019). The empirical payoffs in our case will be network performance metrics when defence protocols are simulated under adversarial conditions. Analysis of these games will ultimately allow us to identify which strategies are robust across the space of possible threats.

While we can test protocols against a range of threats, EGTA also enables us to explore how different defence configurations perform under adversarial conditions. In this context, a strategy is not just a protocol, but a protocol instantiated with a specific set of configuration parameters. We define the configuration space in advance, guided by domain expertise, practical constraints, and knowledge of plausible operating ranges. For example, when evaluating threshold-based protocols, the configuration space might

include threshold values centred around a realistic operating point. This structured exploration allows us to identify not just which protocols are robust, but which configurations of those protocols are the most resilient across a wide range of threat models. This approach directly addresses the concern raised in Section 5.1.1.1, where traditional evaluations do not sufficiently explore the configuration space.

We formally introduce our normal form empirical game as a tuple, $\Gamma(N, S, U)$. Where:

- $N = \{a, d\}$ is the set of types (or players) in the game where a is the attacker and d the defender,
- S is the cartesian product of player strategies, $S_a \times S_d$, and
- U is the set of utility functions, or payoffs, for all players, $U = \{u_a, u_d\}$, where $u_i : S \to \mathbb{R}$, $i \in N$.

u_a and u_d can theoretically take any performance metric produced as a result of the simulations, where the defender wishes to increase this metric and the attacker aspires to do the opposite. This dichotomy is represented as a zero sum, non-cooperative game (Fujiwara-Greve, 2015), meaning $\sum_{i \in N} u_i(s_a, s_d) = 0, \forall s_a, \forall s_d$. These utility values are approximated using samples derived from seeded simulation runs. Formally, a *sample* is a performance metric $u_i^m(s_a, s_d)$ recorded for player i under a specific strategy profile in a single seeded simulation run, m. A *strategy profile*, defined as $s = (s_d, s_a)$, is a pair of strategies chosen by each player. Θ denotes the set of all such samples across all strategy profiles and all seeded runs. To construct a game, we define a mapping $\mathbb{M}(\Theta)$ that selects, aggregates, and averages the values associated with each strategy profile to produce empirical utility values. The resulting empirical game is denoted by $\Gamma = \mathbb{M}(\Theta)$.

As all attack strategies are tested against all defence strategies, we can envisage a matrix of performance metrics where each element is the simulation result of a specific strategy profile. A complete set of payoffs for all strategy profiles can be represented by a *payoff matrix*, where the rows represent S_a and the columns represents S_d.

Since this is an empirical game, a strategy profile must produce observable effects in the FANET that influence the final performance metrics. Selecting which performance metrics to include within the evaluation requires careful consideration. For example, a pure intrusion detection system would not directly affect the packet delivery ratio, but would generate detection results such as true positives or false positives. In such cases, it would be more appropriate to use a metric like the F1-score, rather than the packet delivery ratio, to evaluate performance.

When an empirical game is constructed, standard solution concepts such as Nash equilibrium (Fudenberg and Tirole, 1991) can be applied to identify equilibrium

strategy profiles. Previous studies in empirical game-theoretic analysis have successfully used this approach in a range of cybersecurity settings (Prakash and Wellman, 2015; Wellman et al., 2019). However, several limitations make Nash equilibrium unsuitable for our context.

Firstly, Nash equilibrium assumes that both players are fully rational and will always select the best response to the opponent's strategy. This assumption is problematic in FANET environments, where attackers may behave inconsistently, suboptimally or irrationally due to incomplete information or unpredictable intent. Secondly, Nash equilibrium is only guaranteed to exist when players are allowed to mix strategies, i.e select their strategy based on some probability distribution over their strategy set (Fujiwara-Greve, 2015). This form of strategy mixing is not practical in FANETs, where nodes cannot feasibly store and switch between multiple protocol implementations in real time. Most importantly, Nash equilibrium does not provide a way to compare or rank individual strategies. It identifies stable outcomes but offers no insight into how other strategies perform when facing different attacker behaviours. In our setting, this is a critical limitation, as the goal is not simply to find equilibrium but to evaluate and prioritise defence strategies based on their robustness across a range of possible threats.

To overcome these issues, we restrict our analysis to pure strategies, which align more closely with operational constraints in FANET deployments and allow for clearer interpretation. Rather than seeking equilibrium, our objective is to rank defence strategies according to how well they guard against uncertain or adversarial attacker behaviour. This motivates the use of regret-based decision theory, which focuses on minimising the worst-case performance loss relative to the best possible strategy.

In particular, we adopt the minimax regret decision rule (Savage, 1951), which quantifies the maximum shortfall between a chosen strategy and the best alternative that could have been selected in hindsight. The regret for a defending strategy, s_d, is defined as:

$$r_d(s_d) = \max_{s_a \in S_a} \left[\max_{\bar{s}_d \in S_d} u_d(\bar{s}_d, s_a) - u_d(s_d, s_a) \right]. \tag{5.1}$$

This formulation enables us to produce a ranked list of defending strategies, with the top choice being the one that minimises regret across all attacker responses.

5.2.2 Bootstrapping Regret

We mentioned previously that these payoffs (u_i) are averages of the same strategy profile across all simulation seeds. Using the average within the evaluation fails to take into account the variance associated with these measurements. In Section 5.1.1.3, we

referred to this problem as performance uncertainty. In our case, this problem becomes more challenging as we want to estimate how this performance uncertainty affects regret. From Equation 5.1, we can see that regret is sensitive to the differences in performance metrics across strategy profiles. Therefore, if one were to use the techniques from Section 5.2.1 as is, strategy profiles with a large performance variance but a small average regret can be given a favourable ranking.

The issue of measuring performance uncertainty in EGTA settings *a posteriori* has been addressed in several works (Wiedenbeck et al., 2014; Jecmen et al., 2020; Omidshafiei et al., 2019). However, they all focus on gathering more samples with the intent of bounding the regret value in Nash equilibrium profiles. In our case, we fix the number of samples we obtain from the simulations, as our task is to acquire the most informative evaluation from the limited samples that are available.

Similar to a previous work by Wiedenbeck et al. (2014), we choose to implement Bootstrapping (Davison and Hinkley, 1997), a statistical resampling method which creates multiple sampled sets by resampling the original sample set many times. These sampled sets are then averaged as normal to produce a sampled payoff matrix. This way, the variance in performance metrics are reflected in the sampled averages, which are subsequently reflected in the regret values for each defence strategy. This creates a distribution of regret values per strategy, where one regret value corresponds to a sampled payoff matrix brought about by bootstrapping the original payoff matrix. To acquire a single regret value from a distribution, we take the 95th percentile, which represents a worst case regret outcome and also takes into account the spread of the data. This value is then used to rank the defence protocols from lowest regret to highest regret.

5.2.3 Summary

In this section, we shall describe the three step process which allows us to calculate our bootstrapped regret value. Firstly, an alternative sample set $\tilde{\Theta}$, equal in size to the original sample set Θ, is generated by sampling with replacement from Θ. This step implements the bootstrapping technique introduced in Section 5.2.2. Secondly, a game is constructed, $\Gamma = \mathbb{M}(\tilde{\Theta})$, using the approach outlined in Section 5.2.1. Thirdly, we apply Equation 5.1 to determine the maximum regret per defending strategy. Steps 1 through 3 are then repeated B times, with each iteration storing the maximum regret value for each strategy. This produces a distribution of maximum regrets per s_d. From this distribution, we extract the value corresponding to the 95th percentile and order these values from smallest to largest to form our defence strategy ranking. Figure 5.1 describes this process.

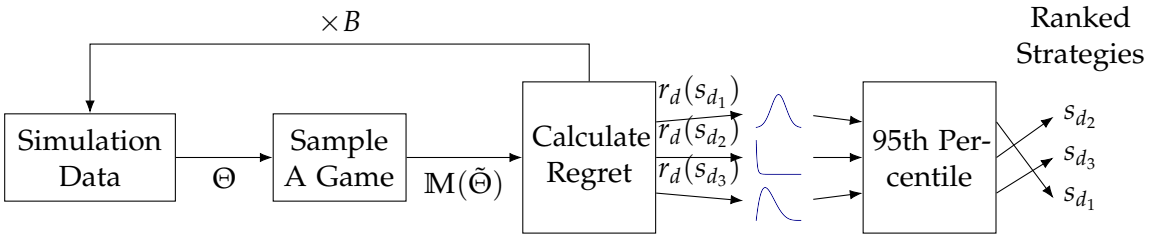


FIGURE 5.1: This diagram summarises the FANET-Rank process, including the simulation of performance metrics, game formation, bootstrapping and extracting the final value from the regret distributions of each strategy.

5.2.4 Practicalities

FANET-Rank has been designed to be simulator-agnostic and, as such, has the ability to processes comma-separated values (CSV) files containing environment and configuration parameters as well as performance metrics. In addition, numerous columns of performance metrics can be appended to these CSV files, all of which can be processed in a similar way. One way of appending more performance data would be to transform previous data into another form. For example, we apply a formula to columns of False Positive, False Negative, True Positive and True Negative values and generate a new column named F1-Score. We are then free to use F1-Score as a utility within FANET-Rank.

While FANET-Rank offers a flexible and extensible evaluation framework, there are some limitations to consider, three of which are discussed below.

Firstly, the simulation environment does not model detailed physical-layer effects such as multipath propagation, signal interference, or obstacle-induced fading. Capturing these aspects would require a significantly more advanced simulator capable of handling 3D trajectory planning, physical object modelling, and UAV collision detection—features that fall outside the scope of this work. Our focus is on evaluating grey hole attacks from a networking and protocol-level perspective, where such physical detail is abstracted.

Secondly, as with many empirical game-theoretic methods, FANET-Rank may face scalability challenges as the number of strategy profiles increases, particularly when combining multiple protocols with varying configuration parameters. Although the framework is designed to support such complexity, the computational cost of generating and evaluating large empirical games remains a practical constraint.

Thirdly, FANET-Rank relies on the presence of both benign and malicious nodes, who have contrasting interests. This assumption enables the comparison of competing strategies under adversarial conditions, but it also constrains the framework's applicability. Specifically, FANET-Rank is not intended for use with protocols that do not involve adversarial behaviour or that operate in entirely cooperative environments.

While this limits its scope, it aligns with the primary focus of FANET-Rank: to assess protocol performance in contested and security-sensitive FANET scenarios.

FANET-Rank has been implemented in Python3.11 and uses common libraries such as Pandas and Matplotlib. Our use of Matplotlib enables researchers to clearly see the regret profiles of each of their strategies, facilitating thoughtful discussions and robust conclusions. In addition, we have also made use of Pandas' pivot table functionality to easily transform CSV data into payoff matrices for use in our EGTA module. Full source code related to this project is publicly available (Hutchins et al., 2024b) and includes an example dataset used to generate the results in this article.

5.3 Assessment Procedure

To assess FANET Rank, we run simulations which use the system model and threat model defined in Chapter 3. Section 5.3.1 defines three defence protocols, each representing a different approach to detecting grey hole behaviour. In assessing FANET-Rank, we choose our defence protocols to ensure a diverse set of behaviours, where some defences are strong against certain attacks but weak against others. This diversity is essential for revealing how effectively FANET-Rank differentiates between these behaviours. If one protocol clearly outperforms the rest, it can obscure these differences and reduce the insight that FANET-Rank is designed to provide.

In Section 5.3.2, we provide an overview of the attack and defence strategies included in this work. And, to provide a baseline for comparison, we include a conventional evaluation method, described in Section 5.3.3, which reflects how performance is typically assessed in existing work.

5.3.1 Protocols Under Evaluation

In this section, we describe our three trust-based defence protocols which we evaluate using our proposed framework. We chose trust based methods due to their continuing popularity in developing secure routing protocols, as indicated in Section 2.9.5. Firstly, Section 5.3.1.1 describes the lightweight protocol and Section 5.3.1.2 describes the Dempster-Shafer Theory (DST) protocol, which uses DST to intelligently accumulate indirect trust values. Lastly, the Velocity-threshold protocol is described in Section 5.3.1.3. This protocol modifies the watchdog mechanism so that next hop nodes are only monitored when their velocity is under a certain value. For all protocols, we also introduce a short form name in bold where the "[X]" place holders represent a particular value of a configuration parameter.

There are two procedures which influence the workings of the defence protocols described in this section. The first procedure allows a defence node to *reject* another node, meaning, all route requests and route replies which are sent to or received from the rejected node are disregarded, preventing the formation of routes with the rejected node. If the rejecting node currently holds a valid route that includes a previously rejected node, the protocol behaves as if the rejected node has disconnected and triggers the transmission of an RERR message.

The three defence protocols implement a base level of shared functionality which is not included as part of the traditional AODV protocol. Most importantly of which, the trust management system from Section 2.9.1.2 is integrated to each protocol to provide data regarding the packet forwarding operations of the next hop node.

5.3.1.1 The Lightweight Protocol

The lightweight protocol (short form: **Lite[X]W[X]**) strongly resembles the CONFIDANT protocol by Buchegger and Le Boudec (2004), which only communicates negative recommendations. We name this the lightweight protocol due to its relatively simple recommendation system compared to the other protocols in this article. In more detail, when a node receives a negative recommendation from a trustworthy node, the indirect trust is halved, with T^{IN} taking on an initial value of 1. In general, this exponential decay puts more weight on the first few recommendations received. In addition, the lightweight protocol implements g and ω as a configuration parameter, as described in Section 2.9.1.4.

5.3.1.2 The Dempster-Shafer Theory (DST) Protocol

The recommendation consolidation module of our Dempster-Shafer theory (DST) protocol (short form: **DST[X]W[X]**), strongly resembles the work of Wei et al. (2014) with minor alterations to introduce node rejection. Dempster-Shafer theory is used in this protocol specifically as a recommendation system to amalgamate indirect trust values. It was first proposed by Chen and Venkataramanan (2005), and since then, has been widely used by other academics in the field (Yang et al., 2014; Wei et al., 2014; Zhao et al., 2012; Chatterjee et al., 2010; Mukherjee et al., 2018; Uma Rani et al., 2022). Similar to the lightweight protocol from Section 5.3.1.1, the DST protocol implements g and ω as a configuration parameter.

5.3.1.3 The Velocity-threshold Protocol

The Velocity-threshold protocol (short form: **Mvel[X]T[X]**) uses the same trust and recommendation systems from the lightweight protocol, however, it also features a

speed detection mechanism which identifies the node's current speed. This speed value is compared with a fixed speed value, $\bar{m}$, to determine if promiscuous mode is activated. As the speed of the node contributes to the accuracy of promiscuous mode, our hypothesis is that monitoring at lower node speeds (in this case, speeds lower than $\bar{m}$) gives better forwarding accuracy, resulting in a more accurate malicious node detection module. It is well known that increased node mobility negatively impacts the reliability of watchdog mechanisms due to the frequent disconnections between nodes (Ramphull et al., 2021; Wu et al., 2013). In contrast to the two previous protocols, ω is a fixed parameter while g is implemented as a configuration parameter. This is because we wanted to investigate how the speed detection mechanism influences the protocol, rather than examining the influence of ω.

5.3.2 Strategy Formation

The defending player can use the lightweight, DST or Velocity-threshold protocol. Moreover, the player can also implement the regular AODV protocol, "Noop". Throughout a single simulation, the protocols and configuration parameters, i.e. the strategies, are fixed for both types. We set $\gamma_f = 1.0$ and $\gamma_d = 0.9$, with a lower value for γ_d indicating that nodes gradually forget bad behaviour over time (see section 2.9.1.2 for more information). We appreciate that these static parameters are defence protocol specific and therefore, could be included as a configuration parameter. In our case, we decided not to vary these parameters as we wanted to test how the variables of the recommendation system and the ratio of direct to indirect trust impacts the performance of the Lightweight and DST protocol. In the case of the Velocity-threshold protocol, we wanted to focus on the speed threshold and the ratio of direct and indirect trust. Of course, FANET-Rank allows the user to vary the fixed parameters if such an analysis is needed. All defence strategies from the lightweight, DST and Velocity-threshold protocols are included as part of the defence strategy set, S_d. The standard AODV protocol (referred to as the Noop or "no operation" strategy) is included in both S_a and S_d.

The attacking player can use the Probabilistic, Time-Based or Precursor-Based attack strategies introduced in Section 3.4.2, as well as the "Noop" strategy. These strategies are included as part of the attack strategy set, S_a. For the attack protocol, we set t_w to 5 seconds, indicating that, for the time-based grey hole attack, drop windows have a length of 5 seconds. In the same vein as the defence protocols, we chose 5 seconds as an example, but other values can be chosen if the user wishes to test their defence protocols with longer or shorter time-based grey hole attacks.

Table 5.2 summarises the defence and attack protocols used in this study, along with their corresponding configuration parameters. Configuration parameter values for each

TABLE 5.2: Strategy Groups

Type	Protocol	Short Form	Configuration Parameters
Attacker	Probabilistic	**PD–**p	p = 0.2, 0.4, 0.6, 0.8, 1.0
Attacker	Time-based	**Tdrp–**b	b = 0.3, 0.5, 0.7
Attacker	Precursor-based	**Sele–**q	q = 0.3, 0.5, 0.7
Defender	DST	**DST-**g**-W-**ω	g = 0.1, 0.2, 0.3 ω = 0.3, 0.5, 0.7
Defender	Lightweight	**Lite-**g**-W-**ω	g = 0.2, 0.3, 0.4 ω = 0.5, 0.7
Defender	Velocity-threshold	**Mvel-**$\bar{m}$**-T-**g	$\bar{m}$ = 5, 10, 15 g = 0.2, 0.3 ω = 0.8
Both	Standard AODV	**Noop**	

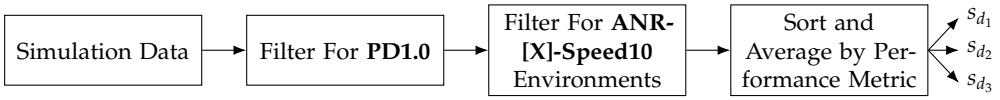


FIGURE 5.2: This figure shows the stages of a typical evaluation method

protocol were chosen based on prior experience and to provide an interesting contrast with the other protocols tested as part of the evaluation.

5.3.3 A Typical Evaluation Methodology

We report the results of a typical evaluation strategy, using our simulation data, to enable a comparison between the original FANET-Rank method and more conventional evaluation procedures. Generating such typical results presents notable challenges, primarily due to incomplete evaluation details in the literature. For instance, the maximum node speed within these evaluations is not always specified. To address this, we adopt a value of 10m/s, as this value was used by several studies (Wei et al., 2014; Bounouni et al., 2022; Sadayan and Ramaiah, 2022). Similarly, the packet dropping rate is often unclear, but some works (Bounouni et al., 2022; Ryu and Kim, 2023) assume a value of 100%, which we adopt as an illustrative case.

Our analysis revealed that the top-performing strategy varied across different ANRs. Therefore, we selected the strategy that performed the best or joint best across all ANRs at a maximum speed of 10m/s. In summary, we report the most frequent top-performing strategy for DPDR in the **ANR-37.5-Speed-10**, **ANR-25-Speed-10**, and **ANR-12.5-Speed-10** environments where the attack strategy is **PD1.0**. Figure 5.2 describes this typical evaluation process.

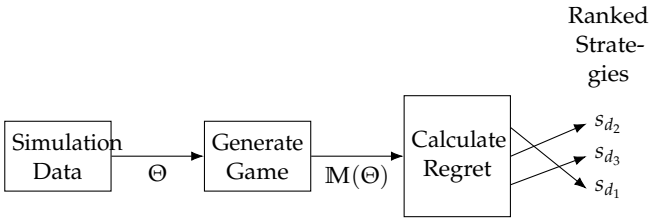


FIGURE 5.3: This figure summarises the stages of FANET-Rank (without bootstrapping applied)

5.4 Experimental Evaluation

This section is split into five distinct parts. Section 5.4.1 compares a non-bootstrapped version of FANET-Rank, named the *original* FANET-Rank, with typical evaluation procedures. This comparison highlights the advantages of using regret as a ranking based metric, which considers the entire threat landscape. In Section 5.4.2, we compare original FANET-Rank with the bootstrapped version and show how the bootstrapping module can affect strategy rankings by considering the variation in node placement and velocity. In section 5.4.3, we will highlight how configuration parameters within a single protocol make a substantial difference to their ranking, which demonstrates the need to include configuration parameters within the evaluation process. In Section 5.4.4, we show how varying both ANR and maximum node speed has an effect on the strategies chosen. And finally, in Section 5.4.5, we will use FANET-Rank to compare and contrast the protocols using three different metrics: DPDR, EED and F1-Score, defined in Section 2.6. This section will analyse the game from the attacker's perspective by ranking the best attack strategies.

5.4.1 Test Case 1: Ranking Across the Threat Landscape

FANET-Rank's strategy selection may appear unfavourable at first glance, however, when examining the graphs more closely, we will see significant gains by choosing FANET-Rank rather than on typical strategies. Initially, we shall describe the first three stages of original FANET-Rank, depicted in Figure 5.3, comparing the strategies generated with conventional evaluation methodologies.

In Figure 5.4, we plot the first place original FANET-Rank strategy, **Mvel05T0.3**, with the typical strategy **Lite0.2W0.7** in environment **ANR-37.5-Speed-10**. The DPDR of strategy **Lite0.2W0.7** clearly outperforms strategy **Mvel05T0.3** when considering the rightmost set of attack strategies. However, what is not considered is the DPDR loss if the attacker were to issue a probabilistic grey hole attack with 0.6 drop rate (**PD0.6**). Here, we can observe more significant losses with strategy **Lite0.2W0.7** compared with strategy **Mvel05T0.3**. This example showcases the fundamental change in perspective which is needed in the evaluation of these protocols: we are not trying to identify the

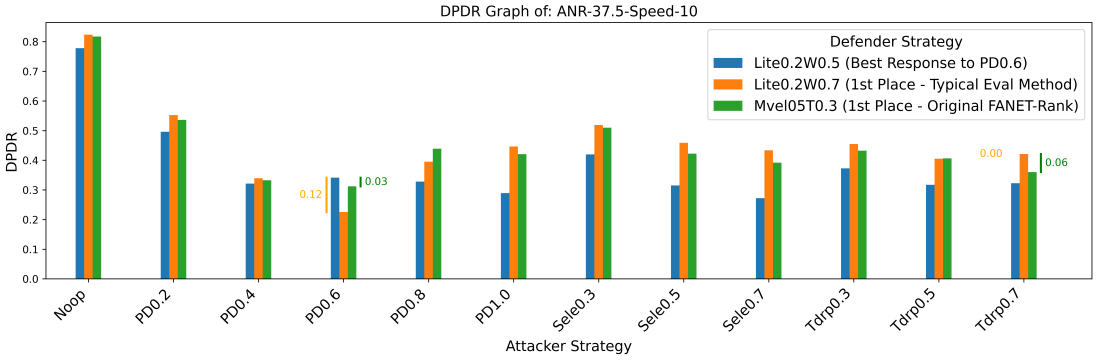


FIGURE 5.4: This graph shows the DPDR across the threat landscape for three defence strategies. **Mvel05T0.3** is the strategy selected by FANET-Rank, while **Lite0.2W0.7** is selected by a typical evaluation method. The best performing defence strategy (**Lite0.2W0.5**), when only considering the attacking strategy **PD0.6**, is also displayed. The difference between the top performing strategy and both **Lite0.2W0.7** and **Mvel05T0.3** is displayed for **PD0.6** and **Tdrp0.7**.

TABLE 5.3: Top 7 FANET-Rank Results of DPDR.
Environment: 25% ANR, Max Speed 5m/s

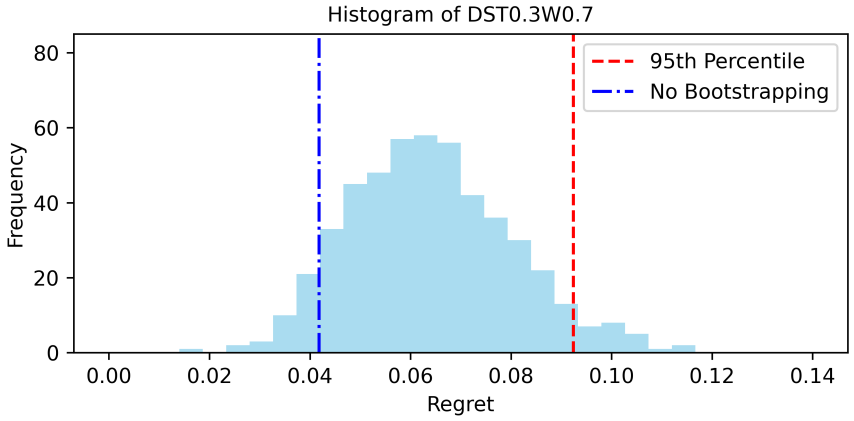
defence Strategy	95th Percentile (2 d.p)
Lite0.3W0.5	0.09
Lite0.3W0.7	0.10
Mvel10T0.3	0.15
Mvel05T0.3	0.15
Mvel15T0.3	0.15
Lite0.4W0.7	0.20
Lite0.2W0.5	0.22

best performing protocol under one specific attack, rather, we are trying to identify the protocol which performs more consistently across the threat landscape we have defined. This way, the attacker cannot exploit protocols which do not perform well under one specific grey hole attack.

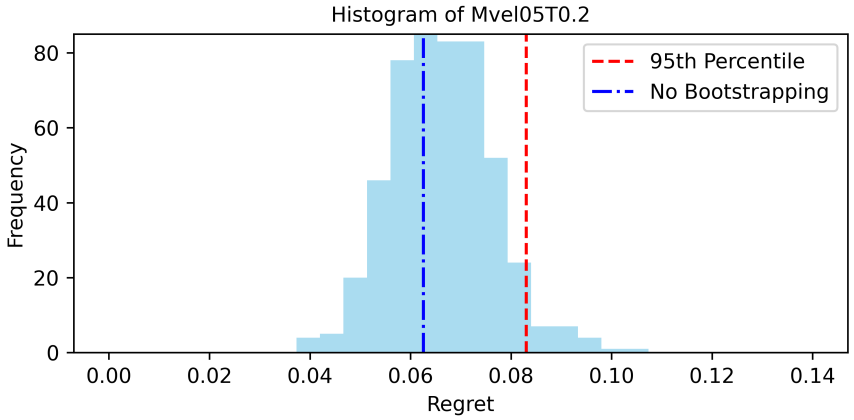
For **Lite0.2W0.7**, the maximum regret experienced was 0.116, an almost 12% loss in DPDR at **PD0.6**. In contrast, the maximum regret value for **Mvel05T0.3** was 0.061, which is only a 6% loss in DPDR at **Tdrp0.7**. These losses in DPDR value are compared against the best performing defence strategy for that attack strategy, i.e. the *best response*. For example, at **PD0.6**, the best defence strategy was **Lite0.2W0.5**, so the DPDR difference between **Lite0.2W0.7** and **Lite0.2W0.5** is 0.116. These regret values are also indicated in Figure 5.4.

5.4.2 Test Case 2: Bootstrapping

In this section we describe the impact that boostrapping has on the selection of strategies. Contrary to the previous section, we will not compare typical evaluation



(A) Regret Distribution of **DST0.3W0.7**



(B) Regret Distribution of **Mvel05T0.2**

FIGURE 5.5: These graphs detail the regret distribution of two strategies, **DST0.3W0.7** in Figure 5.5a and **Mvel05T0.2** in Figure 5.5b. As a reminder, a strategy with a lower regret value is more advantageous. The blue lines (or "-." lines) show the average value from the samples, without bootstrapping. The red lines (or "-" lines) show the 95th percentile from the regret distribution (the bars). Notice how the strategies would be ordered if one were to take the regret value indicated by the blue set of lines over the regret value indicated by the red set of lines. This demonstrates how bootstrapping influences the strategy rankings.

procedures with FANET-Rank, rather, original FANET-Rank (Figure 5.3) with full FANET-Rank (Figure 5.1). For the bootstrapping module, we set $B = 500$, indicating that we construct regret distributions from 500 sampled games. Let us consider Figure 5.5, which shows the regret distributions from two strategies from the **ANR-12.5-Speed-25** environment. The blue line (indicated by the "-." line) in both figures represents the output of the original game, i.e. the output of original FANET-Rank, while the red line (indicated by the "-" line) represents the 95th percentile of the regret distributions, which are based on full FANET-Rank. The rankings of these strategies are dependent on which set of lines are chosen (red or blue). It is clear that if we choose to rank these strategies with the red line (FANET-Rank), a more representative strategy ranking would be obtained, as FANET-Rank takes into account

TABLE 5.4: 1st Place Defence Strategies Per Environment (DPDR)

Attack Node Ratio Max Node Speed	12.5%	25%	37.5%
5	**Lite0.3W0.5**	**Lite0.3W0.5**	**Lite0.3W0.5**
10	**Mvel05T0.3**	**Mvel05T0.3**	**Mvel05T0.3**
15	**DST0.3W0.7**	**Mvel10T0.3**	**Mvel05T0.3**
25	**Mvel05T0.2**	**Mvel15T0.3**	**Mvel05T0.3**
35	**DST0.3W0.7**	**Mvel05T0.2**	**Mvel10T0.3**

the distribution of regrets. In contrast, producing a ranking with the blue lines ignores the regret variations seen in other games, which can produce surprisingly different regret values. Overall, we observe that the application of bootstrapping can adjust regret values by up to 5% as indicated by the histogram of **DST0.3W0.7** in Figure 5.5.

5.4.3 Test Case 3: Strategy Configuration

Throughout our experiments, we also observed how influential the configuration parameters are when comparing different environments. Specifically, we found it interesting to observe the ranking tables themselves and identify the differences in regret with identical protocols and different configuration parameters. For example, Table 5.3 shows the results of FANET-Rank in the **ANR-25-Speed-5** environment; notice the significant DPDR regret difference (10%) between **Lite0.3W0.7** and **Lite0.4W0.7**. So much so, that one may have disregarded the protocol entirely if **Lite0.3W[X]** had not shown promise.

5.4.4 Test Case 4: Evaluation in a Diverse Range of Environments

Table 5.4 presents the 1st place ranked strategy for each environment. Notably, a consistent pattern emerges: in environments with speeds of 5m/s and 10m/s, strategies remain consistent across all ANRs. However, as we increase the speed to 15m/s and beyond, we observe variations in strategies and sometimes even protocols across different ANRs. This provides evidence to suggest that assessing these defence protocols by altering both speed and ANR provides greater insight compared with typical evaluation strategies which only evaluate in environments which alter ANR or speed.

5.4.5 Protocol Evaluation: Example Use Case

This section focuses on a real life use case by using FANET-Rank to compare the protocols using three performance metrics: DPDR, EED and F1-Score. With this use

case, we will show that FANET-Rank not only provides effective rankings, but also allows the user to explore the advantages and disadvantages of their protocols.

In addition, another useful aspect of FANET-Rank is the ability to frame the problem from the attacking side, i.e. create regret distributions for each attacking strategy, $s_a \in S_a$, choose the 95th percentile and rank from lowest to highest value. This yielded a mix of expected and intriguing results which will be described at the end of each metric section.

Referring to Table 5.4, the **Mvel[X]T[X]** protocol consistently secures the 1st place rank in 10 out of 15 environments, with various configuration parameters playing pivotal roles. Conversely, the **DST[X]W[X]** protocol appears better suited for lower ANR environments, possibly due to the higher defending node density, which facilitates trust information exchange among neighboring nodes. This claim is substantiated by the observed value of 0.7 as an indirect trust weighting. Another discernible trend is the increasing velocity threshold and/or the decrease in sensitivity in **Mvel[X]T[X]** protocols with increasing maximum speeds in the environment. It seems that the inclusion of a velocity threshold to limit the amount of direct trust information is beneficial compared with the other protocols. However, the actual velocity chosen makes a noticeable difference to the performance of the network. Therefore, if we are allowed to assume that the drones know the maximum speed of the network, it may be advantageous to define a threshold parameter based on a fraction of the maximum speed.

5.4.5.1 DPDR

We have previously mentioned that reducing the intensity of the attack can actually increase its effectiveness. This hypothesis is substantiated by our results, where the vast majority of 1st placed attack strategies have intensity in the middle range (Table 5.5). Moreover, Figure 5.4 also shows this trend between the values of **PD0.2** and **PD1.0**, as **PD0.6** has a lower DPDR than both **PD0.4** and **PD0.8** in two out of the three protocols shown. One notable exception is the **Tdrp-0.7** strategy, which is the most aggressive time-based grey hole attack. We believe the advantage of this attack strategy lies in its ability to constantly oscillate between exhibiting attacking and benign behavior. In low ANR environments, this confusing behaviour would target the recommendation systems, making them less likely to report a node as malicious.

5.4.5.2 EED

Thus far, we have primarily referenced DPDR as the performance metric of choice. This is because it remains the most ubiquitous metric in this research field. However, in the

TABLE 5.5: 1st Place Attack Strategies Per Environment (DPDR)

Attack Node Ratio / Max Node Speed	12.5%	25%	37.5%
5	**PD0.6**	**PD0.6**	**PD0.6**
10	**PD0.6**	**PD0.6**	**PD0.6**
15	**PD0.6**	**PD0.6**	**PD0.6**
25	**Tdrp0.7**	**PD0.6**	**PD0.6**
35	**Tdrp0.7**	**PD0.6**	**PD0.6**

TABLE 5.6: 1st Place Attack Strategies Per Environment (EED)

Attack Node Ratio / Max Node Speed	12.5%	25%	37.5%
5	**PD1.0**	**PD0.8**	**PD1.0**
10	**Sele0.5**	**PD0.8**	**Sele0.7**
15	**Tdrp0.5**	**Tdrp0.7**	**Sele0.5**
25	**Tdrp0.5**	**PD0.8**	**PD0.8**
35	**PD1.0**	**PD0.8**	**PD1.0**

rest of this section, we shall describe our findings from our FANET-Rank evaluation where EED is the metric of interest. Overall, we found that the "Noop" strategy is 1st place in the vast majority of environments. This result is not surprising, as AODV is optimised to find the quickest and shortest route. Any process which interferes with the node selection is likely to increase the EED. From the attack perspective, we find a much more varied selection of attack strategies, as indicated by Table 5.6. In general, these strategies tend to be a lot more aggressive, with parameter values of 0.8 and 1.0 featuring heavily. These findings support the notion that different attack strategies can impact different QoS parameters, i.e. some are much more likely to degrade EED than DPDR.

5.4.5.3 F1-Score

Table 5.7 show the results from F1-score. Interestingly, we would expect Table 5.7 and 5.4 to have high correlation, as a high classification accuracy should translate to a high DPDR. However, We find that the **Lite[X]W[X]** protocol dominates Table 5.7 with mostly different configuration parameters from what was seen in Table 5.4. High value monitoring threshold parameters feature heavily in Table 5.7, indicating that more sensitive systems are better at attack node classification than increasing the DPDR of the network.

From the attacking perspective, the attacker wishes to decrease the F1-score, thereby reducing the accuracy with which the defence protocols can correctly identify an attacking node. These results are presented in Table 5.8 and, unsurprisingly, if the

TABLE 5.7: 1st Place Defence Strategies Per Environment (F1-Score)

Attack Node Ratio / Max Node Speed	12.5%	25%	37.5%
5	**Lite0.3W0.5**	**Lite0.4W0.7**	**Lite0.2W0.5**
10	**Lite0.3W0.5**	**Lite0.4W0.7**	**Lite0.4W0.7**
15	**Mvel05T0.3**	**Lite0.4W0.7**	**Lite0.4W0.7**
25	**Mvel05T0.3**	**Lite0.3W0.5**	**Lite0.4W0.7**
35	**Mvel05T0.3**	**Lite0.3W0.5**	**Lite0.2W0.5**

TABLE 5.8: 1st Place Attack Strategies Per Environment (F1-Score)

Attack Node Ratio / Max Node Speed	12.5%	25%	37.5%
5	**Noop**	**Noop**	**Noop**
10	**Noop**	**PD0.2**	**Noop**
15	**Noop**	**Noop**	**Noop**
25	**Noop**	**Noop**	**Noop**
35	**Noop**	**Noop**	**Noop**

attacker were to do nothing, i.e. **Noop**, the defence protocols find it difficult to identify any attack. This predictable, yet unremarkable outcome, reinforces the perspective shared in Section 5.4.5.2, emphasizing that the choice of the most appropriate attack depends on the intentions of the attacker.

5.5 Conclusion

We have created an evaluation framework named FANET-Rank and analysed its effectiveness in evaluating protocols. We assessed FANET-Rank by comparing it to a typical evaluation method across four test cases. Additionally, we also provided an example use case which gave useful insights. FANET-Rank selects protocols based on the performance observed from the entire threat landscape as well as considering environmental factors such as the randomness associated with node placement and speed. In addition, it also considers a broader set of environment parameters and defence configuration parameters compared with previous evaluation methods.

Our assessment of FANET-Rank demonstrated several key advantages over typical evaluation methods. First, FANET-Rank effectively accounts for the attacker's strategy in defence strategy selection. This was evident in our results, where the highest ranking strategy under a typical evaluation method resulted in a 12% drop in DPDR under another attack. Due to this, FANET-Rank identified and demoted this strategy. Second, we showed that bootstrapping the regret metric yields more reliable rankings, with observed changes of up to 5% in regret values per strategy. Third, we highlighted

FANET-Rank's sensitivity to configuration tuning, where a small change in parameters led to a 10% improvement in performance. Fourth, we demonstrated that varying both the ANR and the maximum node speed simultaneously, rather than in isolation, had a significant impact on the strategies selected. This result shows that considering only one of these factors is insufficient, and that meaningful evaluation requires both to be varied together. Finally, we illustrated how FANET-Rank can be applied with multiple utility metrics (DPDR, EED, and F1-score) to gain insight into protocol behaviour from both the defender's and attacker's perspectives. This allowed us to capture both well established and more subtle trends across a wide range of environments.

We encourage other researchers in this field to use FANET-Rank as their protocol evaluation method of choice. By adopting FANET-Rank, researchers can confidently report their protocol's performance compared with other protocols.

Chapter 6

A Mobility Conditioned Direct Trust Mechanism For Reliable Packet Forwarding Prediction

The previous chapters established two key foundations for defending against grey hole attacks in flying ad hoc networks (FANETs): first, that such attacks produce temporal signatures that can be detected early (Chapter 4); and second, that defence protocols should be evaluated under diverse conditions using a rigorous game-theoretic framework (Chapter 5). Having established how attacks can be identified and how protocols can be assessed, the logical next step is to design a defence mechanism that directly addresses the shortcomings of existing approaches. This chapter therefore focuses on direct trust (introduced in Section 2.9.1.2). Direct trust is a widely used defence mechanism, however, it suffers from some critical limitations.

Most direct-trust approaches estimate next-hop reliability using aggregated forwarding outcomes, a measure we call the observed forwarding ratio (oFR). However, oFR misattributes many mobility-induced drops as malicious behaviour and ignores the time gap between trust updates and forwarding decisions. To address these issues, this chapter introduces a mobility-conditioned direct trust (MCDT) mechanism. MCDT is trained using the FAN-GHETS24 dataset, and its effect on network performance is rigorously assessed with the FANET-Rank framework.

This chapter is organised as follows. Section 6.1 outlines the motivation for MCDT and its novel contributions. Section 6.2 describes the two main components of MCDT: the packet scoring model (PSM) and the packet forwarding prediction model (PFPM). Finally, Section 6.3 presents an experimental evaluation using FANET-Rank, demonstrating that MCDT outperforms existing direct trust approaches in terms of DPDR and EED in the majority of environments.

6.1 Introduction

The oFR is obtained during the information-gathering stage of trust management, typically through the watchdog mechanism described in Section 2.9.1.1. In prior work, oFR has often been treated directly as a measure of trust, or used as a feature within larger trust-computation models. For example, the expected value of the beta distribution introduced in Section 2.9.1.2 can be interpreted as an oFR or embedded in a broader predictive framework.

However, using oFR values blurs the distinction between raw observations and the conclusions that should be derived from them, leading to trust estimates that may not account for the underlying causes of packet loss. This chapter argues for a separation of concerns: oFR is a descriptive statistic reflecting what has been observed while predicted forwarding rate (pFR) should take into account the node's mobility when each packet was sent as well as node's mobility at decision time. The limitations of using oFR are substantiated in Section 6.1.1 which are subsequently addressed in Section 6.1.2 by introducing MCDT.

As in Chapter 4, we define the *eval node* as any benign node in the network and the *subject node* as any other node within the eval node's communication range that is being evaluated.

6.1.1 Motivation

As well as being a predictive measure of past packet forwarding performance, ρ is also lightweight to store and calculate, since only α and β need to be updated (Equation 2.5). It is important to note that modern approaches to direct trust computation still use the standard ρ measurement as an oFR (Tropea et al., 2024; Singh and Verma, 2020; Barka et al., 2018; Uma Rani et al., 2022). Moreover, some works augment or fuse oFR with other direct information sources to provide a more accurate value. For example, Singh and Verma (2020) introduces TBCS, which uses Fuzzy Logic to combine signal strength, transmission delay, node energy, oFR, and past interactions to determine a direct trust value. Barka et al. (2018) use a scaled version of ρ, but do not combine this with any other direct trust information before the trust aggregation stage.

However, these works do not take into account the circumstances under which this information is collected. The observed forwarding ratio (oFR) only reflects the number of packets forwarded or dropped between two nodes. While this is a useful performance indicator, it is not a reliable predictor of future behaviour. Treating oFR directly as a predicted forwarding ratio (pFR) means that two nodes with an unstable link will be assigned a lower trust value, even when the subject node's forwarding intentions are entirely benign.

Ryu and Kim (2023) implement an oFR using the distances between the subject node and the nodes in the remaining route. However, in their update equation, they set the reputation update to 0 if the packet was not forwarded by the subject node. This still indicates a significant oversight in direct trust updates: if the distance between the eval and subject nodes is large, and the subject node is benign, the reputation of the node would still decrease due to environmental factors, not malicious intent.

Mobility-related variations in oFR are only part of the challenge. A more fundamental issue is that prior studies which use trust management often overlook the distinction between when information is collected and when it is actually required. In other words, Trust management depends not only on the accuracy of information, but also on whether it is gathered, computed, and applied at the appropriate stage in the protocol. So far, this section has argued that packet context (i.e., mobility information) should be collected at packet forwarding time and used at trust computation time. From previous works, it is unclear when trust information gathering is collected.

Moreover, we must consider the time at which we make the decision to trust or distrust a certain node. The decision stage of trust management often occurs at a different time than the trust computation stage, which means that the trust value applied in decision-making does not necessarily reflect the moment when the direct trust update was last performed. For example, let us assume a subject node is benign and has successfully forwarded many packets for the evaluation node. Now, let us also assume that the same node is later chosen as the next hop when the link quality is poor. In this case, the resulting pFR should be lower, even though the oFR remains high.

6.1.2 Contributions

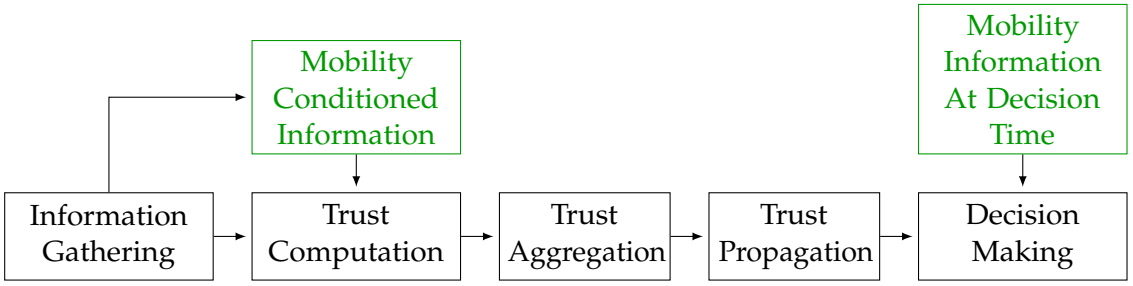


FIGURE 6.1: This figure shows the improved trust management process. The boxes in green indicate two pieces of additional information applied during the trust management process: mobility-conditioned information applied prior to trust computation, and mobility information applied during decision making.

To remedy the issues presented in Section 6.1.1, this chapter proposes the **M**obility-**C**onditioned **D**irect **T**rust (MCDT) mechanism, which consists of two models trained from the FAN-GHETS24 dataset: a packet scoring model (PSM) and packet forwarding prediction model (PFPM). The PSM uses mobility-conditioned information

to assess the maliciousness of next hop nodes. In addition, the PFPM applies current mobility metrics to each queued packet, ensuring that forwarding decisions are based on up-to-date information. These models are practical to train and deploy, since the required mobility information can be gathered at any point during the trust management process. The updated trust management process is displayed in Figure 6.1, where the green boxes indicate the additional stages that this work proposes.

The effectiveness of MCDT is evaluated through simulation in the FANET described in Section 3. The proposed approach is compared against a baseline that uses a popular oFR, the expectation of the beta distribution, introduced in Section 2.9.1.2. Rather than comparing forwarding ratios directly, the evaluation focuses on the impact of each method on network performance. In particular, DPDR and EED are used as performance indicators, and results are reported across a range of trust thresholds.

6.2 Methodology

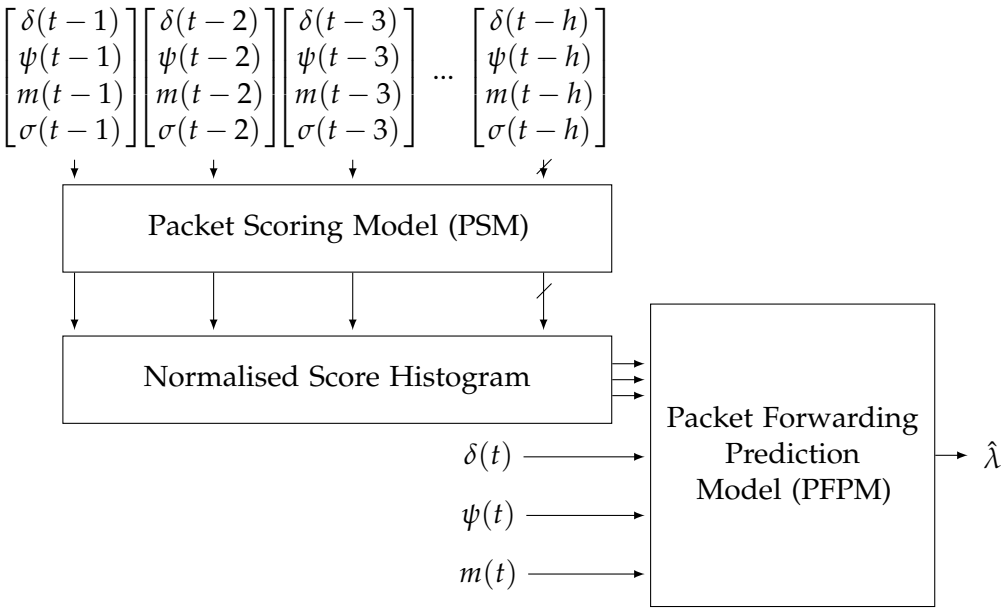


FIGURE 6.2: This diagram summarises the movement of data in our mobility conditioned direct trust (MCDT) mechanism

Our mobility-conditioned direct trust (MCDT) mechanism consists of two models trained with FAN-GHETS24. As stated previously, our goal is to better predict the packet forwarding rate (pFR) by taking into account mobility related data. This model can then be used as a drop-in replacement for direct trust mechanisms and can be combined with indirect trust values to produce an overall trust value.

The Packet Scoring Model (PSM) is detailed in Section 6.2.1 and produces a score per forwarded packet which indicates how benign the interaction with the subject node was.

If the score is high, this would indicate that the interaction with this node bares a similar resemblance to interactions with benign nodes in FAN-GHETS24. The PSM acts on all historic and completed packet interactions and uses the packet's distance (δ), direction (ψ), speed (m) and forwarding status (σ) to predict how benign the interaction was. We say that a packet interaction is completed when the packet is confirmed as either forwarded by the next hop node or dropped due to some other factor.

The packet forwarding prediction model (PFPM) is detailed in section 6.2.2 and takes a normalised score distribution from multiple historic forwarding events and combines this with the current distance ($\delta(t)$), direction ($\psi(t)$) and speed ($m(t)$) measurements to produce a predicted forwarding rate, $\hat{\lambda}$. Note that we do not have the forwarding status, $\sigma(t)$, as this particular packet has not been sent yet.

In MCDT, trust estimation is shaped by the complementary roles of both immediate mobility context and historical forwarding evidence, each influencing the trust value in distinct but interdependent ways. Historical evidence—captured through the PSM encodes how reliably a neighbour has forwarded packets under the mobility conditions present at the time of each transmission, allowing the system to build a long-term behavioural profile that distinguishes malicious patterns from noise or short-term fluctuations. This historical component is essential for recognising persistent grey hole behaviour, but it alone cannot account for situations where packet drops are caused by transient mobility factors such as rapid separation, directional divergence, or high-speed manoeuvres. Immediate mobility therefore plays a critical corrective role: the PFPM incorporates current distance, direction, and speed to adjust trust estimates at decision time, ensuring that drops likely caused by mobility are not misinterpreted as malicious. In practice, historic evidence provides stability and resilience against sporadic anomalies, while immediate mobility provides contextual sensitivity that prevents false positives. MCDT's strength arises from balancing these two sources: history governs long-term trust trends, whereas immediate mobility governs short-term trust accuracy, together enabling the mechanism to reliably differentiate genuine grey hole behaviour from mobility-induced packet loss. Figure 6.2 is a diagrammatic representation of MCDT and shows how the PSM and PFPM modules interact to produce the final pFR value, $\hat{\lambda}$. h represents the number of historic confirmed packets.

6.2.1 Packet Scoring Model (PSM)

We use 50% of the FAN-GHETS24 sequences and filter the data to include only benign node interactions. For each confirmed data packet in these sequences, we record four values: distance (δ), direction (ψ), speed (m), and forwarding status (σ). The forwarding status is a boolean that indicates whether the packet was successfully forwarded by the subject node. Because each MTS (see Section 4.3) already contains the full interaction history, including promiscuously detected packets, the forwarding status can be

determined directly from the sequence. As described in Section 2.9.1.1, a packet is marked as forwarded if the eval node observes the correct TTL, Packet ID, and origin node. If the correct packet is not observed within the specified timeout period, the packet is marked as dropped and the forwarding status is set accordingly. With this procedure applied to every data packet and every available sequence, we can build a regression dataset, as defined below.

Definition 6.1 (Regression Dataset)**.** Given a collection of feature vectors, $\mathbf{X} = \langle x^{(1)}, x^{(2)}, \ldots, x^{(n)} \rangle$, and their associated continuous-valued targets, $\mathbf{Y} = \langle y^{(1)}, y^{(2)}, \ldots, y^{(n)} \rangle$, a regression dataset is defined as

$$\mathbf{D} = \langle \langle x^{(i)}, y^{(i)} \rangle \mid x^{(i)} \in \mathbb{R}^k,\ y^{(i)} \in \mathbb{R},\ i = 1, 2, \ldots, n \rangle,$$

where k is the dimensionality of the feature space and n is the number of data points.

For the PSM dataset, $\mathbf{D}_{\text{PSM}}$, each data point is represented by a feature vector of length $k_{PSM} = 3$ and the number of data points within the dataset was found to be $n_{PSM} = 258,685$. Specifically, each data point is represented by a feature vector, $x^{(i)} = (d, \psi, m)$, and has an associated target, $y^{(i)} = \sigma$. The aim of the training process is to find a function which maps packet mobility information to forwarding status for benign nodes $\mathcal{B} : \mathbb{R}^{k_{PSM}} \to \mathbb{R}$.

As indicated by the dataset's name, this task is formulated as a regression problem rather than a probabilistic one. This design choice is intentional, since empirical evidence showed that probabilistic models tended to be overly conservative, producing outputs clustered around 0.5. By adopting a regression formulation, the model is encouraged to generate more decisive scores for each packet, rather than defaulting toward uncertainty.

To implement the PSM, we use a feed-forward neural network with the architecture described in Table 6.1. The model has two outputs: one produces a score associated with packet dropping and the other produces a score associated with packet forwarding. During inference, the observed outcome of the packet transmission determines which output is selected, and the corresponding score is then used as the scoring measure for that packet.

Input Dim	Output Dim	Description
4	32	Linear + BatchNorm1d + ReLU
32	16	Linear + BatchNorm1d + ReLU
16	2	Linear + Sigmoid

TABLE 6.1: network architecture for the Packet Scoring Model (PSM).

In inference, the trained PSM acts on every completed packet transaction, yielding a per-node distribution of scores. To provide this to the PFPM (Introduced in Section

6.2.2) in a fixed format, the scores are converted into a normalised histogram with three bins: $[0, 0.33)$, $[0.33, 0.66)$, $[0.66, 1.0]$. This produces a three-element input for the PFPM model, $(c^{(0)}, c^{(1)}, c^{(2)})$, where $\sum_i c^{(i)} = 1$. This method simplifies the storage requirements of each node, since the node only needs to maintain a count per bin rather than a large set of individual scores.

6.2.2 Packet Forwarding Prediction Model (PFPM)

Using the next 40% of the FAN-GHETS24 dataset, we filter all MTSs (of all types) by choosing a point in the sequence where a data packet is to be sent to the subject node. Specifically, this data packet indicates when a new route has been formed and the first data packet is being sent using this new route. This is called the *evaluation point* and can be identified by observing the Packet IDs in sequence and identifying the specific places where the Packet ID stops incrementing by 1 with each packet sent. In this process, we make sure there are at least γ confirmed packet interactions before the evaluation point and at least γ packets are sent using this new route. The data associated with the confirmed packets, (d, ψ, m), is fed into the PSM model from Section 6.2.1 and a histogram is generated from the scores. In addition, the current distance ($\delta(t)$), direction ($\psi(t)$) and speed ($m(t)$) values of the subject node are also used as an input to the PFPM to ensure that current information is used at decision time.

As the outcome of this particular data packet is known, and all subsequent packets which result from this interaction are also known, we can generate a known forwarding rate, λ. This data can be used to train a model which generates a pFR.

To create the PFPM regression dataset, $\mathbf{D}_{\text{PFPM}}$, data is generated using the methodology described above. Subsequently, this data includes the following fields: $(c^{(0)}, c^{(1)}, c^{(2)}, \delta(t), \psi(t), m(t), \lambda)$ where the last field, λ, is used as the target parameter for training. The $c^{(0)}$, $c^{(1)}$ and $c^{(2)}$ variables represent the normalised score count bins described at the end of Section 6.2.1. This means that k_{PFPM} is 6 and when generating the dataset, n_{PFPM} was found to be approximately 6000. The learning objective is to find a function, $\mathcal{P} : \mathbb{R}^{k_{\text{PFPM}}} \to \mathbb{R}$, which predicts $\hat{\lambda}$.

Similar to PSM, the PFPM is a standard neural network model with an architecture described in Table 6.2. However, the PFPM model only has a single output to represent $\hat{\lambda}$ and takes 6 parameters as inputs.

Input Dim	Output Dim	Description
6	32	Linear + BatchNorm1d + ReLU
32	16	Linear + BatchNorm1d + ReLU
16	1	Linear + Sigmoid

TABLE 6.2: Network architecture for the Packet Forwarding Prediction Model (PFPM).

6.3 Experimental Evaluation

In this section, γ takes on a value of 5, which indicates that at least 5 packets are needed before and after the evaluation point in order to gather the data needed for the PFPM. This value was empirically selected to balance two competing effects. A larger threshold excludes many samples and thus shrinks the dataset. By contrast, a smaller threshold produces more observed forwarding rates of 0 or 1, eliminating intermediate cases and skewing the training. The same γ value is used at both dataset creation stage and evaluation stage to ensure consistency in both training and inference. If γ packets are not confirmed, MCDT is disabled and the default AODV forwarding rules apply, i.e. the packet is always sent if the route is valid.

We compare our direct trust mechanism with the Lightweight protocol (Section 5.3.1.1) and the velocity-threshold protocol (Section 5.3.1.3). The Lightweight protocol uses the expectation of the beta distribution, as described in Section 2.9.1.2. Since the DST protocol employs the same direct trust mechanism as the Lightweight protocol, it is excluded from this evaluation. In contrast, the velocity-threshold protocol is included, as it applies a slightly different approach when calculating direct trust with the beta distribution: packet interactions are only recorded when the current node's speed is below a defined threshold.

To gather results, we use the FANET-Rank evaluation framework from Chapter 5 with a modified lists of defence protocols and configuration parameters. As we are interested in network performance rather than classification, DPDR and EED are used as performance metrics.

While FANET-Rank results are normally displayed as tables of protocols separated by environment, the regret differences between the top performing protocols provides an interesting indication of measurable improvement and tactical advantage. The term "protocol" is used intentionally here, rather than "strategy", as this evaluation highlights the differences between protocol families rather than protocols with particular configuration parameter settings.

This section is split into two parts. Firstly, Section 6.3.1 evaluates MCDT with direct trust mechanisms. Specifically, the direct trust ratio in all protocols is fixed to provide direct trust only, i.e. $\omega = 1.0$. Secondly, Section 6.3.2 evaluates MCDT with direct-indirect aggregated trust mechanisms (defined in Section 2.9.1.4). Ergo, the second evaluation uses node recommendations and direct-indirect trust aggregation to better predict the trust of the subject node.

TABLE 6.3: Strategy Groups For Direct Trust Mechanism vs MCDT Evaluation

Type	**Protocol**	**Short Form**	**Configuration Parameters**
Attacker	Probabilistic	**PD–**p	p = 0.2, 0.4, 0.6, 0.8, 1.0
Attacker	Time-based	**Tdrp–**b	b = 0.3, 0.5, 0.7
Attacker	Precursor-based	**Sele–**q	q = 0.3, 0.5, 0.7
Defender	Mobility-Conditioned Direct Trust*	**MCDT-**g	g = 0.2, 0.3 ω = 1.0
Defender	Lightweight	**Lite-**g**-W-**ω	g = 0.2, 0.3, 0.4 ω = 1.0
Defender	Velocity-threshold	**Mvel-**v**-T-**g	g = 0.2, 0.3, 0.4 v = 5, 10, 15 ω = 1.0
Both	Standard AODV	**Noop**	

*Proposed Mechanism.

TABLE 6.4: 1st Place Defence Strategies Per Environment for DPDR (MCDT vs Direct Trust Mechanism)

Attack Node Ratio Max Node Speed	12.5%	25%	37.5%
5	**MCDT-0.3**	**MCDT-0.3**	**MCDT-0.3**
10	**MCDT-0.2**	**MCDT-0.3**	**MCDT-0.3**
15	**MCDT-0.3**	**MCDT-0.3**	**MCDT-0.3**
25	**MCDT-0.3**	**Lite0.4W1.0**	**Mvel15T0.2**
35	**MCDT-0.3**	**MCDT-0.3**	**MCDT-0.2**

TABLE 6.5: 1st Place Defence Strategies Per Environment for EED (MCDT vs Direct Trust Mechanism)

Attack Node Ratio Max Node Speed	12.5%	25%	37.5%
5	**Noop**	**Noop**	**MCDT-0.2**
10	**MCDT-0.2**	**MCDT-0.3**	**MCDT-0.2**
15	**MCDT-0.3**	**MCDT-0.3**	**MCDT-0.2**
25	**MCDT-0.2**	**MCDT-0.2**	**MCDT-0.3**
35	**MCDT-0.2**	**MCDT-0.2**	**MCDT-0.3**

6.3.1 MCDT vs Direct Trust Mechanisms

Table 6.3 details the protocols and the configuration parameters used in the MCDT vs direct trust mechanism evaluation. It is worth noting that the seeded simulation runs also depend on the software used for the simulation itself. Because the software was recompiled with additional source code for this evaluation, identical random seeds do

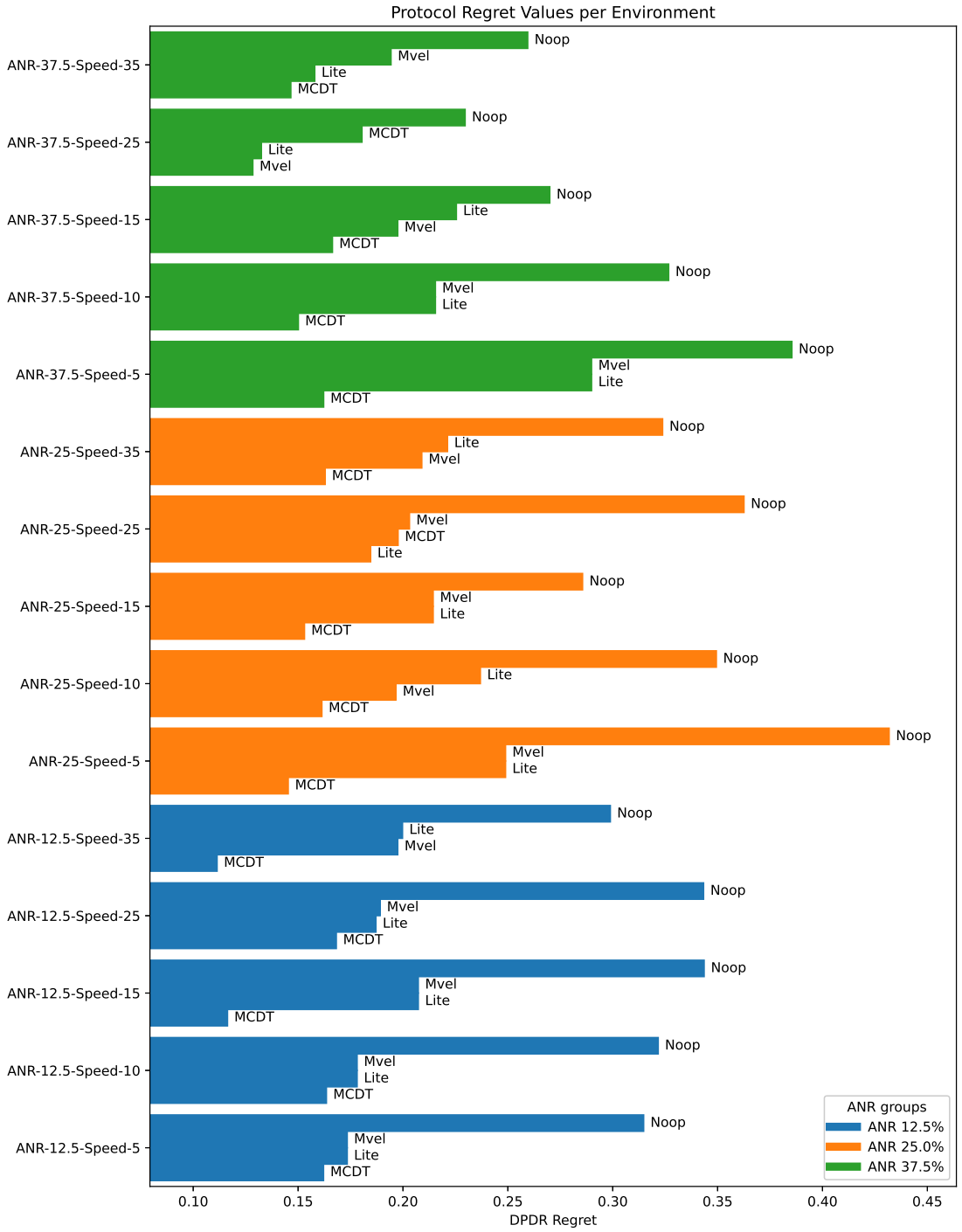


FIGURE 6.3: This figure shows the performance of all four protocols in the evaluation of MCDT versus direct trust mechanisms. Each protocol is represented by the best (lowest) DPDR regret it achieved across all available strategies.

not reproduce the same node speeds and positions compared with the dataset simulations, making this evaluation free from mobility related biases.

Table 6.4 presents the DPDR results for the MCDT vs direct trust mechanism evaluation. For clarity, each value corresponds to the lowest regret obtained across all strategy

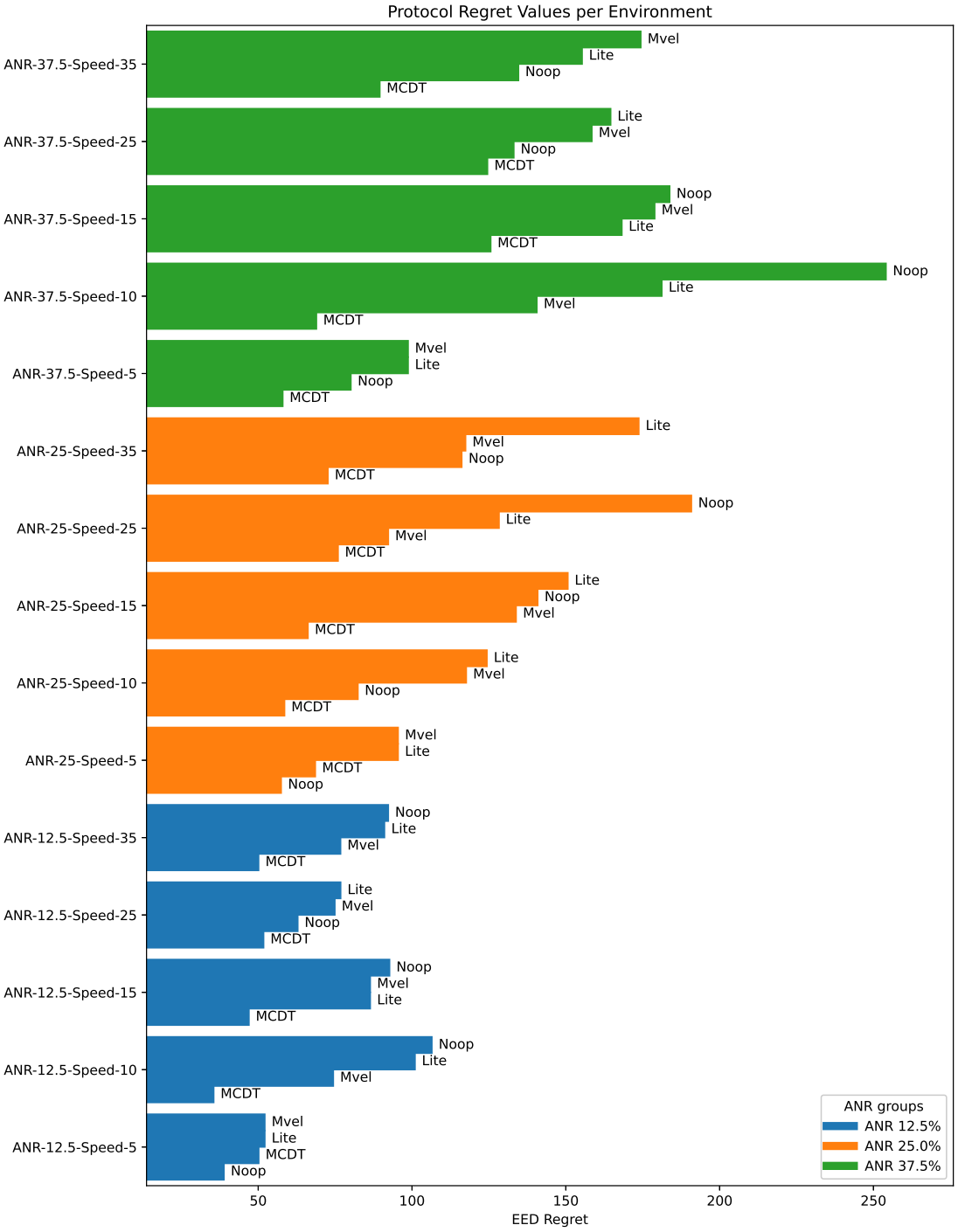


FIGURE 6.4: This figure shows the performance of all four protocols in the evaluation of MCDT versus direct trust mechanisms. Each protocol is represented by the best (lowest) EED regret it achieved across all available strategies.

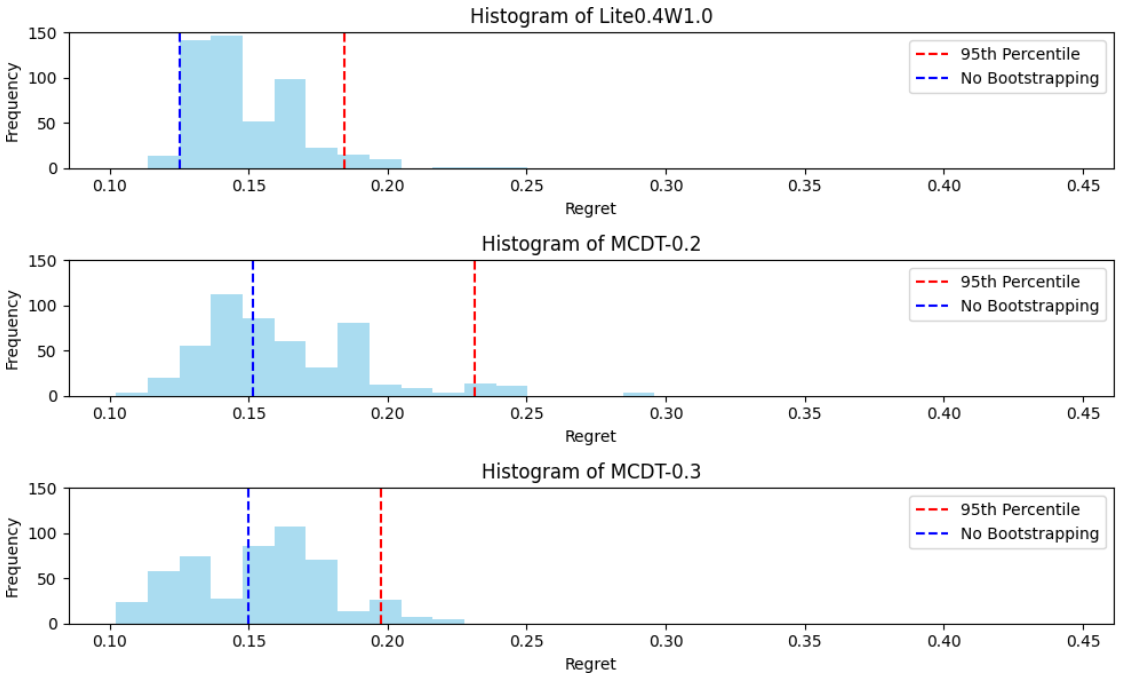


FIGURE 6.5: This graph shows the regret distributions of the **Lite0.4W1.0**, **MCDT-0.2** and **MCDT-0.3** strategies in the **ANR-25-Speed-25** environment.

configurations within the same protocol. MCDT failed to be placed first in only two environments, **ANR-25-Speed-25** and **ANR-37.5-Speed-25**. The following analysis considers these two environments in more detail to better understand the differences observed in their results.

In the **ANR-25-Speed-25** environment, **Lite0.4W1.0** achieved 1st place and **MCDT-0.3** ranked 2nd, with the DPDR regret difference between them being less than 2%. Referring to Figure 6.5, the regret distribution of **MCDT-0.3** appears broader, while that of **Lite0.4W1.0** is slightly more compact. This suggests that in this specific environment, small variations in node mobility or contact duration may have influenced the outcome, leading to marginal performance differences rather than a consistent advantage of one protocol over the other.

In contrast, in the **ANR-37.5-Speed-25** environment, **MCDT-0.2** shows a larger regret difference of approximately 5% from the 1st place strategy, **Mvel15T0.2**. When comparing the scale of Figures 6.5 and 6.6, it is clear that MCDT's regret distributions remain consistent across both environments. Therefore, this result does not indicate a particular weakness in MCDT but rather that **Lite** and **Mvel** align particularly well with the characteristics of the **ANR-37.5-Speed-25** environment, allowing them to achieve slightly better performance under these specific conditions.

Referring to table 6.4, **MCDT-0.3** was the most popular strategy, with 85% of first place MCDT strategies implementing the $g = 0.3$ configuration parameter. Moreover, the bar graph in Figure 6.3 indicates that the difference in DPDR regret between the

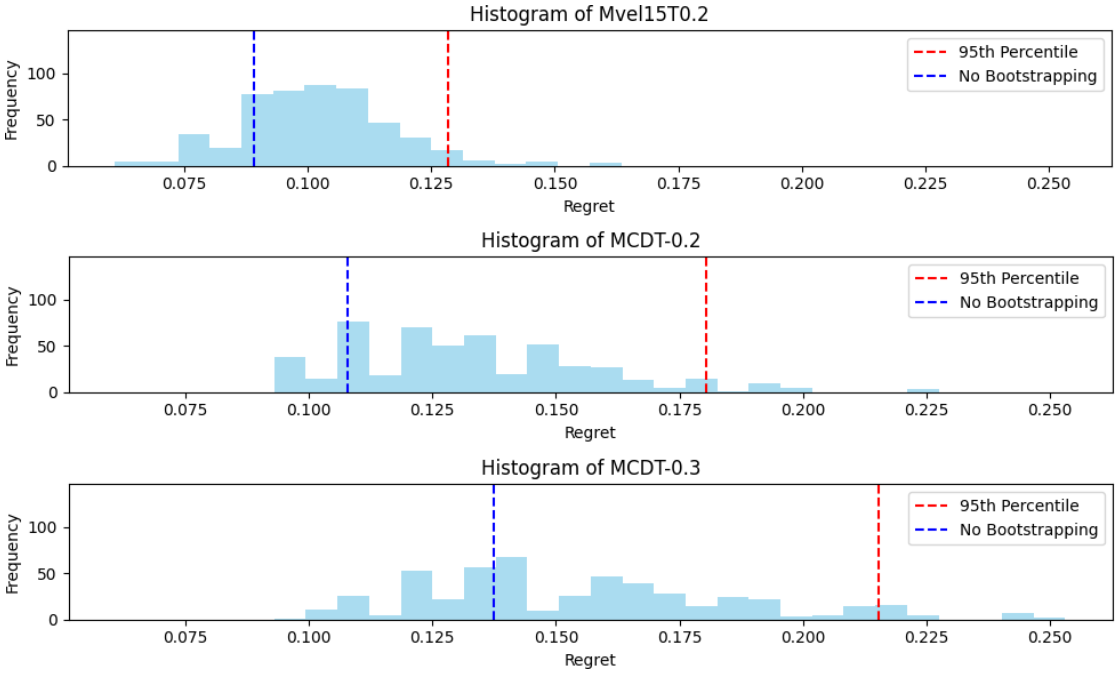


FIGURE 6.6: This graph shows the regret distributions of the **Mvel15T0.2**, **MCDT-0.2** and **MCDT-0.3** strategies in the **ANR-37.5-Speed-25** environment.

Lightweight and MCDT protocol is approximately 3% in the **ANR-25-Speed-25** environment. Furthermore, the largest DPDR difference between MCDT and the next top performing protocol was in the **ANR-37.5-Speed-5** environment, with a DPDR regret difference of approximately 12%. This result presents a large improvement from the lightweight protocol, which implements the beta distribution direct trust mechanism.

Table 6.5 shows the EED results for the comparison between the direct trust mechanism and the MCDT protocol. In this table, MCDT achieves the highest performance in 87% of environments. This is a particularly surprising result as we predicted that deviating from the AODV protocol (which prioritises EED) would result in a larger EED. However, we find that the opposite is true. Moreover, the graph in Figure 6.4 indicates that reductions in EED can reach approximately 75 ms when compared to the next top performing protocol (**ANR-37.5-Speed-10**).

6.3.2 MCDT vs Direct-Indirect Aggregated Trust Mechanisms

MCDT demonstrates strong performance when compared with a direct trust mechanism. To further assess its effectiveness, we also conducted an evaluation against a direct–indirect aggregated trust mechanism. FANET-Rank easily facilitates this, as the direct trust configuration parameter can be adjusted to incorporate recommendations from other UAVs in the network. Table 6.6 presents the strategies used in this second evaluation. These strategies closely resemble the groups defined in Table 5.2, with the

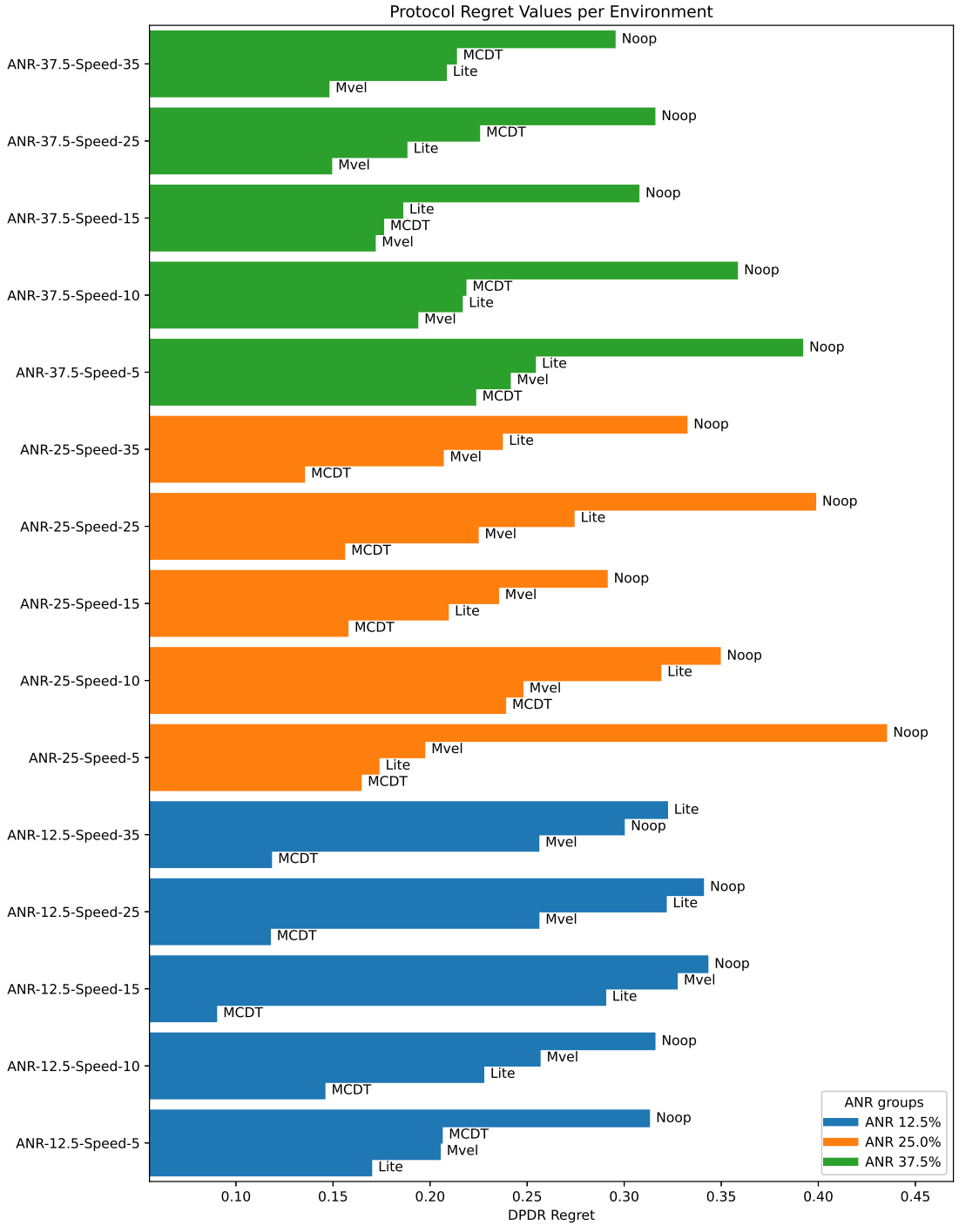


FIGURE 6.7: This figure shows the performance of all four protocols in the evaluation of MCDT versus direct-indirect aggregated trust mechanisms. Each protocol is represented by the best (lowest) DPDR regret it achieved across all available strategies.

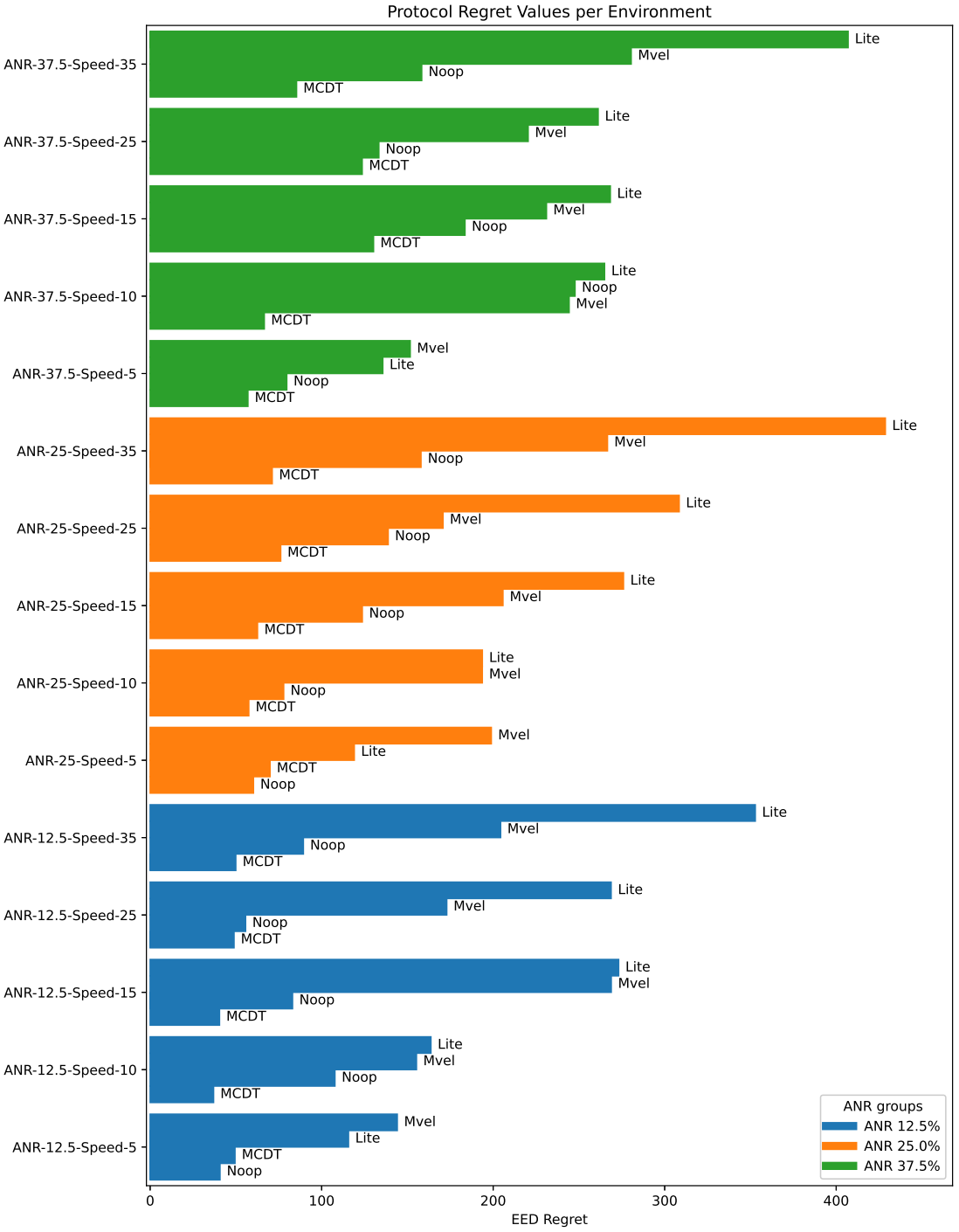


FIGURE 6.8: This figure shows the performance of all four protocols in the evaluation of MCDT versus direct-indirect aggregated trust mechanisms. Each protocol is represented by the best (lowest) EED regret it achieved across all available strategies.

TABLE 6.6: Strategy Groups For Direct-Indirect Aggregated Trust Mechanism vs MCDT Evaluation

Type	**Protocol**	**Short Form**	**Configuration Parameters**
Attacker	Probabilistic	**PD–**p	p = 0.2, 0.4, 0.6, 0.8, 1.0
Attacker	Time-based	**Tdrp–**b	b = 0.3, 0.5, 0.7
Attacker	Precursor-based	**Sele–**q	q = 0.3, 0.5, 0.7
Defender	Mobility-Conditioned Direct Trust*	**MCDT-**g	g = 0.2, 0.3 ω = 1.0
Defender	Lightweight	**Lite-**g**-W-**ω	g = 0.2, 0.3, 0.4 ω = 0.5, 0.7
Defender	Velocity-threshold	**Mvel-**v**-T-**g	g = 0.2, 0.3, 0.4 v = 5, 10, 15 ω = 0.8
Both	Standard AODV	**Noop**	

*Proposed Mechanism.

TABLE 6.7: 1st Place Defence Strategies Per Environment for DPDR (MCDT vs Direct-Indirect Aggregated Trust Mechanism)

Attack Node Ratio Max Node Speed	12.5%	25%	37.5%
5	**Lite0.2W0.7**	**MCDT-0.3**	**MCDT-0.3**
10	**MCDT-0.3**	**Mvel05T0.3**	**Mvel05T0.2**
15	**MCDT-0.3**	**MCDT-0.3**	**Mvel05T0.2**
25	**MCDT-0.3**	**MCDT-0.3**	**Mvel10T0.2**
35	**MCDT-0.3**	**MCDT-0.3**	**Mvel05T0.2**

TABLE 6.8: 1st Place Defence Strategies Per Environment for EED (MCDT vs Direct-Indirect Aggregated Trust Mechanism)

Attack Node Ratio Max Node Speed	12.5%	25%	37.5%
5	**Noop**	**Noop**	**MCDT-0.2**
10	**MCDT-0.2**	**MCDT-0.3**	**MCDT-0.2**
15	**MCDT-0.3**	**MCDT-0.2**	**MCDT-0.2**
25	**MCDT-0.2**	**MCDT-0.2**	**MCDT-0.3**
35	**MCDT-0.2**	**MCDT-0.2**	**MCDT-0.3**

addition of the MCDT protocol and its configuration parameters. All other simulation details are identical to the evaluation in Section 6.3.1.

Table 6.7 presents the DPDR results for the direct-indirect aggregated trust mechanism vs MCDT evaluation. MCDT struggles at higher ANR in this evaluation, most likely because the proportion of malicious nodes increases. In such environments, node recommendations become more important for distinguishing benign from malicious

nodes, meaning that MCDT's reliance on direct trust limits its effectiveness. With this in mind, the result is not surprising, and future work that combines MCDT with a recommendation system could clearly benefit from the direct trust performance gains offered by MCDT. On the other hand, as the ANR decreases, MCDT becomes the superior protocol as indicated by the graph in Figure 6.7. Specifically, the regret difference between MCDT and the next top performing protocol can be as high as 20% (**ANR-12.5-Speed-15**); a substantial improvement compared with other protocols in this evaluation.

Table 6.8 presents the EED results for the direct-indirect aggregated trust mechanism vs MCDT evaluation. With the exception of the **ANR-25-Speed-15**, all of the first place strategies are the same as Table 6.5, indicating that MCDT is a robust choice for preserving EED. Similar to Table 6.5, only two environments (**ANR-12.5-Speed-5** and **ANR-25-Speed-5**) had superior EED regret values. However, The graph in Figure 6.8 shows that MCDT was the next top performing protocol in both environments, with minimal regret difference values of approximately 15ms. The highest gains can be seen in the **ANR-37.5-Speed10** environment with an EED regret value of approximately 175ms.

6.4 Conclusion

These results demonstrate that MCDT is a robust direct trust mechanism compared with the standard AODV protocol (**Noop**), the standard beta distribution direct trust mechanism (**Lite**), and a mobility-conditioned beta distribution mechanism (**Mvel**). In direct trust FANET-Rank evaluations, the results show that MCDT achieves reductions in DPDR regret of up to 12% and in EED regret of up to 75 ms compared with its competitor. In direct–indirect aggregated trust evaluations, the improvements in some environments are even greater, with reductions in DPDR regret of up to 20% and in EED regret of up to 175 ms.

From these results, its interesting to point out the maximum performance gains in direct-indirect aggregated trust evaluations were generally much higher. This may be due to the false positive rate induced by incorrect negative recommendations broadcasted through the network. Although node recommendations help the decision making process of other protocols at higher ANR values, they prove to be detrimental in lower ANR environments, making MCDT superior.

Across both direct and direct-indirect aggregated evaluations, MCDT performs exceptionally well in low ANR environments, achieving strong results in both EED and DPDR. Furthermore, MCDT shows improved EED performance compared with the standard AODV protocol, which is important in practice since FANETs are not always under attack and efficiency during normal operation must also be taken into account.

Adding to this practicality, the combined model size of PSM and PFPM is approximately 20 KB, which can be readily accommodated on small-scale UAVs.

Although MCDT improves trust accuracy by combining historic forwarding evidence with immediate mobility context, it inevitably introduces additional overhead compared with traditional direct-trust mechanisms. This overhead appears in three forms: computational, storage, and communication. Computationally, both the PSM and PFPM must execute lightweight neural network inferences; however, these models were deliberately designed to be compact so that they can run efficiently on resource-constrained UAV hardware without impacting real-time routing operations. Storage overhead arises from maintaining a histogram of past packet-scoring values rather than storing raw interaction histories, which significantly reduces memory usage while still preserving sufficient behavioural structure for prediction. Importantly, MCDT does not introduce any new communication overhead: it relies solely on locally collected watchdog observations and does not exchange recommendations, trust reports, or model updates. As a result, although MCDT requires modest on-board computation, its overall overhead remains low and predictable, making it practical for FANET deployments where bandwidth is scarce and processing resources are constrained.

Chapter 7

Conclusion

This thesis has explored defence protocols for mitigating grey hole attacks (GHAs) in Flying Ad Hoc Networks (FANETs). These mitigation efforts are particularly important given the use of FANETs in mission-critical applications such as search and rescue operations, where maintaining Quality of Service (QoS) is essential even in the presence of GHAs. This chapter summarises the key contributions of this thesis across three sections and outlines directions for future research.

Section 7.1 concludes the work on FAN-GHETS24 and recommends investigating transformer-based model architectures to further improve the Decision Latency and accuracy of grey hole attack classification. Section 7.2 concludes the work on FANET-Rank and highlights the importance of reducing barriers to its adoption within the research community. Thirdly, Section 7.3 concludes the work on the mobility-conditioned direct trust mechanism and advocates the use of model analysis and interpretability tools to evaluate its practicality for real-world deployment. Finally, Section 7.4 suggests some more general future directions which apply to all contributions of this thesis.

7.1 A FANET Dataset for Fast GHA Detection

The first contribution of this thesis introduces FAN-GHETS24, a novel dataset that captures two-node interaction histories as multi-feature time series, each labelled according to different types of grey hole attacks. In contrast, previous datasets in this field did not represent data as time series, instead relying on features averaged over fixed time windows. Furthermore, earlier datasets excluded mobility features inherent to FANETs and did not record pairwise node interactions under varying grey hole attack scenarios. The utility of FAN-GHETS24 was demonstrated through the development of an early classification model, CEC-FANET, which achieved high

accuracy and timeliness in detecting grey hole attacks compared to conventional fixed-time base classifiers.

We encourage other researchers to make use of this dataset. And, with the advent of transformer-based architectures (Vaswani et al., 2017), it would be valuable to investigate how such models could be applied to FAN-GHETS24 to enhance both the accuracy and speed of grey hole attack detection. Specifically, these architectures employ a mechanism called attention to relate different elements within an input sequence by weighting their relevance to one another, enabling the model to capture contextual dependencies regardless of their position in the sequence. This capability is particularly advantageous in grey hole attack classification, where indicators of malicious behaviour may appear intermittently across the time sequence, allowing the model to integrate these dispersed cues into a coherent understanding of the attack pattern.

7.2 A Game-Theoretic Evaluation Framework for FANETs

The second contribution of this thesis presents FANET-Rank, a novel evaluation framework developed to overcome several methodological limitations that can artificially inflate the performance of weaker defence protocols. Specifically, we identified a previously unexplored space of environmental and configuration parameters, and, used game theory in conjunction with bootstrapping to 1) manage performance variability across different threat scenarios and 2) to account for the uncertainty induced by node mobility. Our results demonstrated that FANET-Rank provides a more reliable and discriminative assessment of defence protocols, enabling fairer comparisons and revealing performance inconsistencies that conventional evaluation methods tend to overlook.

Future work on this framework could involve intelligently expanding the sampling process to obtain more accurate regret estimates. However, such complexity may offer diminishing returns in evaluation performance. A more impactful direction may instead lie in enhancing the accessibility of FANET-Rank to lower the barriers to adoption and subsequently improve the consistency of results across studies. Specifically, developing clear documentation, publishing explanatory blog posts, providing cloud-ready deployment scripts, and delivering community talks could help promote wider and more effective use of the framework.

7.3 A Mobility-conditioned Direct Trust Mechanism

The third contribution of this thesis introduces the mobility-conditioned direct trust mechanism (MCDT), which leverages mobility-related information to generate more reliable direct trust estimations. Using the FANET-Rank framework, we evaluated MCDT and found that it achieved superior performance compared to traditional direct trust mechanisms.

Given that this contribution is intended for direct integration within UAVs, further model analysis would be valuable to assess its suitability for real-world deployment. In particular, investigating its energy consumption, model size, and overall feasibility of implementation in operational UAV networks would provide crucial insights into its practical applicability.

7.4 Summary

In summary, this thesis has advanced the field of FANET defence protocols by addressing critical challenges in the detection, evaluation, and mitigation of grey hole attacks (GHAs). The three core contributions collectively enhance both the methodological foundations and the practical applicability of defence mechanisms within FANET environments.

Although the dataset, evaluation framework, and trust mechanism developed in this thesis are grounded in a FANET-specific SAR scenario, their underlying principles are broadly generalisable to other mobile ad-hoc networking contexts. The FAN-GHETS24 dataset captures fundamental interaction dynamics, like mobility-induced link variability, packet-level forwarding behaviour, and temporal patterns of selective dropping, that arise in any highly mobile, decentralised wireless environment. This means that its structure can be adapted to MANETs, VANETs, or multi-robot systems with minimal modification. Likewise, the FANET-Rank framework makes no assumptions about UAV hardware or AODV-specific semantics; its empirical game theoretic treatment of attacker–defender interactions, coupled with bootstrapping to capture performance variability, can be applied to any protocol family where configuration choices and adversarial behaviour jointly influence network performance. Finally, the Mobility-Conditioned Direct Trust Mechanism (MCDT) is built on general trust estimation concepts, combining historical interaction evidence with immediate mobility context. This extends naturally to other domains where distinguishing malicious behaviour from mobility or congestion-induced loss is essential.

While these contributions represent meaningful progress, there remain several opportunities for further improvement. Future research could explore a broader

spectrum of threat types, such as wormhole and modification attacks, to enhance the generality of both datasets and defence mechanisms. Additionally, modelling and incorporating energy consumption would allow for a more comprehensive evaluation of defence protocol efficiency, which is an especially important factor for deployment in energy-constrained FANETs.

Collectively, these directions point toward a future in which FANET defence protocols become not only more accurate and resilient, but also efficient and practical for deployment, enabling safer and more reliable aerial communication in search and rescue applications.

References

Khattab M.Ali Alheeti, Anna Gruebler, and Klaus McDonald-Maier. Intelligent Intrusion Detection of Grey Hole and Rushing Attacks in Self-Driving Vehicular Networks. *Computers 2016, Vol. 5, Page 16*, 5(3):16, 7 2016. ISSN 2073-431X. . URL https://www.mdpi.com/2073-431X/5/3/16/htmhttps://www.mdpi.com/2073-431X/5/3/16.

Anas Alkhatieb, Emad Felemban, and Atif Naseer. Performance Evaluation of Ad-Hoc Routing Protocols in (FANETs). *2020 IEEE Wireless Communications and Networking Conference Workshops, WCNCW 2020 - Proceedings*, 4 2020. .

Iman Almomani, Bassam Al-Kasasbeh, and Mousa Al-Akhras. WSN-DS: A Dataset for Intrusion Detection Systems in Wireless Sensor Networks. *Journal of Sensors*, 2016, 2016. ISSN 16877268. .

Ludovic Apvrille, Tullio Tanzi, and Jean Luc Dugelay. Autonomous drones for assisting rescue services within the context of natural disasters. *2014 31th URSI General Assembly and Scientific Symposium, URSI GASS 2014*, 10 2014. .

Muhammad Yeasir Arafat and Sangman Moh. A Q-Learning-Based Topology-Aware Routing Protocol for Flying Ad Hoc Networks. *IEEE Internet of Things Journal*, 9(3): 1985–2000, 2 2022. ISSN 23274662. .

Ms Avneet, Kaur Saluja, Ms Sweta, A Dargad, and Ms Krupali Mistry. A Detailed Analogy of Network Simulators - NS1, NS2, NS3 and NS4. *International Journal on Future Revolution in Computer Science & Communication Engineering*, 3(12):291–295, 12 2017. ISSN 2454-4248. URL https://www.ijfrcsce.org/index.php/ijfrcsce/article/view/411.

Sorav Bansal and Mary Baker. Observation-based Cooperation Enforcement in Ad Hoc Networks. Technical report, Computer Science Department, Stanford University, 7 2003. URL https://arxiv.org/abs/cs/0307012v2.

Ezedin Barka, Chaker Abdelaziz Kerrache, Nasreddine Lagraa, Abderrahmane Lakas, Carlos T. Calafate, and Juan Carlos Cano. UNION: A Trust Model Distinguishing Intentional and Unintentional Misbehavior in Inter-UAV Communication. *Journal of*

Advanced Transportation, 2018(1):7475357, 1 2018. ISSN 2042-3195. . URL /doi/pdf/10.1155/2018/7475357https://onlinelibrary.wiley.com/doi/abs/10.1155/2018/7475357https://onlinelibrary.wiley.com/doi/10.1155/2018/7475357.

Ilker Bekmezci, Ozgur Koray Sahingoz, and Şamil Temel. Flying Ad-Hoc Networks (FANETs): A survey. *Ad Hoc Networks*, 11(3):1254–1270, 5 2013. ISSN 1570-8705. .

Sihem Benfriha, Nabila Labraoui, Haythem Bany Salameh, and Hafida Saidi. A Survey on Trust Management in Flying Ad Hoc Networks: Challenges, Classifications, and Analysis. *2023 10th International Conference on Software Defined Systems, SDS 2023*, pages 107–114, 2023. .

Omar Bouhamed, Ouns Bouachir, Moayad Aloqaily, and Ismaeel Al Ridhawi. Lightweight IDS For UAV Networks: A Periodic Deep Reinforcement Learning-based Approach. In *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2021. ISBN 9783903176324.

Mahdi Bounouni, Louiza Bouallouche-Medjkoune, Abderrahmane Beraza, and Adel Daoud. Eliminating Selective Dropping Attack in Mobile Ad Hoc Network. *Wireless Personal Communications*, 123(4):3291–3308, 4 2022. ISSN 1572834X. . URL https://link.springer.com/article/10.1007/s11277-021-09289-z.

Justin A Boyan and Michael L Littman·. Packet Routing in Dynamically Changing Networks: A Reinforcement Learning Approach. *Advances in Neural Information Processing Systems*, 6, 1993.

Sonja Buchegger and Jean-Yves Le Boudec. A Robust Reputation System for P2P and Mobile Ad-hoc Networks. In *P2PEcon '04: The Second Workshop on Economics of Peer-to-Peer Systems*, 6 2004.

Armir Bujari, Carlos T. Calafate, Juan Carlos Cano, Pietro Manzoni, Claudio Enrico Palazzi, and Daniele Ronzani. Flying ad-hoc network application scenarios and mobility models. *International Journal of Distributed Sensor Networks*, 13(10):1–17, 10 2017. ISSN 15501477. . URL https://journals.sagepub.com/doi/full/10.1177/1550147717738192.

Ozlem Ceviz, Pinar Sadioglu, Sevil Sen, and Vassilios G. Vassilakis. A novel federated learning-based IDS for enhancing UAVs privacy and security. *Internet of Things*, 31: 101592, 5 2025a. ISSN 2542-6605. . URL http://arxiv.org/abs/2312.04135.

Ozlem Ceviz, Sevil Sen, and Pinar Sadioglu. Distributed Intrusion Detection in Dynamic Networks of UAVs using Few-Shot Federated Learning. *arXiv preprint*, 2025b. .

V. Chandrasekar, V. Shanmugavalli, T. R. Mahesh, R. Shashikumar, Naiwrita Borah, V. Vinoth Kumar, and Suresh Guluwadi. Secure malicious node detection in flying ad-hoc networks using enhanced AODV algorithm. *Scientific Reports 2024 14:1*, 14(1): 1–16, 4 2024. ISSN 2045-2322. . URL https://www.nature.com/articles/s41598-024-57480-6.

Pushpita Chatterjee, Indranil Sengupta, and S. K. Ghosh. A distributed trust model for securing mobile ad hoc networks. *Proceedings - IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, EUC 2010*, pages 818–825, 2010. .

Manish Devendra Chawhan, Kruttika Karmarkar, Gargi Almelkar, Disha Borkar, Kishor D. Kulat, and Bhumika Neole. Identification and prevention of Gray hole attack using IDS mechanism in MANET. *International Conference on Emerging Trends in Engineering and Technology, ICETET*, 2022-April, 2022. ISSN 21570485. .

Thomas M. Chen and Varadharajan Venkataramanan. Dempster-Shafer theory for intrusion detection in ad hoc networks. *IEEE Internet Computing*, 9(6):35–41, 11 2005. ISSN 10897801. .

Jin Hee Cho, Ananthram Swami, and Ing Ray Chen. A survey on trust management for mobile ad hoc networks. *IEEE Communications Surveys and Tutorials*, 13(4):562–583, 12 2011. ISSN 1553877X. .

A. Chriki, H. Touati, Hichem Snoussi, and Farouk Kamoun. FANET: Communication, mobility models and security issues. *Computer Networks*, 163:106877, 11 2019. ISSN 1389-1286. .

Coptrz. DJI Matrice 300 RTK Spotlight - Secrets & Lies. *https://coptrz.com/blog/secrets-lies-dji-matrice-300-rtk/*, 2024. URL https://coptrz.com/blog/secrets-lies-dji-matrice-300-rtk/.

A. C. Davison and D. V. Hinkley. *Bootstrap Methods and their Application*. Cambridge University Press, 10 1997. ISBN 9780521573917. . URL https://www.cambridge.org/core/books/bootstrap-methods-and-their-application/ED2FD043579F27952363566DC09CBD6A.

DJI. DJI MATRICE 300 RTK Specifications. *https://enterprise.dji.com/matrice-300/specs*, 2024. URL https://enterprise.dji.com/matrice-300/specs.

Irin Dorathy and M. Chandrasekaran. Simulation tools for mobile ad hoc networks: a survey. *Journal of Applied Research and Technology*, 16(5):437–445, 2018. ISSN 2448-6736. . URL https://jart.icat.unam.mx/index.php/jart/article/view/739.

Maryam Faraji-Biregani and Reza Fotohi. Secure communication between UAVs using a method based on smart agents in unmanned aerial vehicles. *Journal of Supercomputing*, 77(5):5076–5103, 5 2021. ISSN 15730484. . URL https://link.springer.com/article/10.1007/s11227-020-03462-0.

Drew. Fudenberg and Jean. Tirole. *Game Theory*. MIT Press, 1991. ISBN 978-0-262-06141-4.

Takako Fujiwara-Greve. *Non-Cooperative Game Theory*, volume 1 of *Monographs in Mathematical Economics*. Springer Japan, Tokyo, 2015. ISBN 978-4-431-55644-2. . URL https://link.springer.com/10.1007/978-4-431-55645-9.

Ning Gao, Le Liang, Donghong Cai, Xiao Li, and Shi Jin. Coverage Control for UAV Swarm Communication Networks: A Distributed Learning Approach. *IEEE Internet of Things Journal*, 9(20):19854–19867, 10 2022. ISSN 23274662. .

Mohamed Ghalwash, Vladan Radosavljevic, and Zoran Obradovic. Early diagnosis and its benefits in sepsis blood purification treatment. *Proceedings - 2013 IEEE International Conference on Healthcare Informatics, ICHI 2013*, pages 523–528, 2013. .

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT Press, 2016. ISBN 9780262337373.

Shashi Gurung and Siddhartha Chauhan. A novel approach for mitigating gray hole attack in MANET. *Wireless Networks*, 24(2):565–579, 2 2018. ISSN 15728196. . URL https://link.springer.com/article/10.1007/s11276-016-1353-5.

Abdulfatai Shola Hanafi, Yakub Kayode Saheed, and Micheal Olaolu Arowolo. An Effective Intrusion Detection in Mobile Ad-hoc Network Using Deep Belief Networks and Long Short-Term Memory. *International Journal of Interactive Mobile Technologies (iJIM)*, 17(19):123–135, 10 2023. ISSN 1865-7923. . URL https://online-journals.org/index.php/i-jim/article/view/27663.

Bryan Harris and Michael Pooler. Record Brazil floods kill 95 and cause $1bn damage, 2024. URL https://www.ft.com/content/dec816f8-1f53-4c47-939c-874efb8218c6.

Saad M. Hassan, Mohd Murtadha Mohamad, and Farkhana Binti Muchtar. Advanced Intrusion Detection in MANETs: A Survey of Machine Learning and Optimization Techniques for Mitigating Black/Gray Hole Attacks. *IEEE Access*, pages 1–1, 9 2024. ISSN 21693536. .

Nima Hatami and Camelia Chira. Classifiers with a reject option for early time-series classification. *Proceedings of the 2013 IEEE Symposium on Computational Intelligence and Ensemble Learning, CIEL 2013 - 2013 IEEE Symposium Series on Computational Intelligence, SSCI 2013*, pages 9–16, 2013. .

Charles Hutchins, Leonardo Aniello, Enrico Gerding, and Basel Halak. MANET-Rank: A Framework for Defence Protocols against Packet Dropping Attacks in MANETs. In *IEEE/IFIP Network Operations and Management Symposium*, 5 2024a.

Charles Hutchins, Leonardo Aniello, Enrico Gerding, and Basel Halak. MANET-Rank Gitlab Repository, 2024b. URL https://git.soton.ac.uk/ch1u20/manet-rank.

Charles Hutchins, Leonardo Aniello, Enrico Gerding, and Basel Halak. A flying ad-hoc network dataset for early time series classification of grey hole attacks. *Scientific Data*, 12(1):1–22, 12 2025. ISSN 20524463. . URL https://www.nature.com/articles/s41597-025-05560-1.

Inspired Flight. IF1200 Specifications. *https://www.inspiredflight.com/if1200-specs.php*, 2024. URL https://www.inspiredflight.com/if1200-specs.php.

International Organization for Standardization. Open Systems Interconnection — Basic Reference Model, 1994.

Roslan Ismail and Audun Josang. The Beta Reputation System. In *BLED 2002 Proceedings*. AIS Electronic Library, 2002. URL http://databases.si.umich.edu/reputations/.

Hassan Jari, Ali Alzahrani, and Nigel Thomas. A Novel Indirect Trust Mechanism for Addressing Black hole Attacks in MANET. *DIVANet 2021 - Proceedings of the 11th ACM Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications*, pages 27–34, 11 2021. . URL https://dl.acm.org/doi/10.1145/3479243.3487296.

Steven Jecmen, Arunesh Sinha, Zun Li, and Long Tran-Thanh. Bounding Regret in Empirical Games. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04): 4280–4287, 4 2020. ISSN 2374-3468. . URL https://ojs.aaai.org/index.php/AAAI/article/view/5851.

Rutvij H. Jhaveri, Sankita J. Patel, and Devesh C. Jinwala. DoS attacks in mobile ad hoc networks: A survey. *2nd International Conference on Advanced Computing and Communication Technologies, ACCT 2012*, pages 535–541, 2012. .

Gihani Jinarajadasa, Lakmal Rupasinghe, and Iain Murray. A Reinforcement Learning Approach to Enhance the Trust Level of MANETs. *2018 National Information Technology Conference, NITC 2018*, 11 2018. .

Matthew Johnston, Claudiu Danilov, and Kevin Larson. A Reinforcement Learning Approach to Adaptive Redundancy for Routing in Tactical Networks. *Proceedings - IEEE Military Communications Conference MILCOM*, 2019-October:267–272, 7 2018. .

Jinsi Jose and Deepa V Jose. Deep learning algorithms for intrusion detection systems in internet of things using CIC-IDS 2017 dataset. *International Journal of Electrical and Computer Engineering (IJECE)*, 13(1):1134–1141, 2023. ISSN 2088-8708. .

Saeed Kaviani, Bo Ryu, Ejaz Ahmed, Kevin A. Larson, Anh Le, Alex Yahja, and Jae H. Kim. Robust and Scalable Routing with Multi-Agent Deep Reinforcement Learning

for MANETs. *arXiv:2101.03273 [cs]*, 1 2021. URL https://arxiv.org/abs/2101.03273v2.

Muhammad Fahad Khan, Kok Lim Alvin Yau, Rafidah Md Noor, and Muhammad Ali Imran. Routing Schemes in FANETs: A Survey. *Sensors 2020, Vol. 20, Page 38*, 20(1):38, 12 2019. ISSN 1424-8220. . URL https://www.mdpi.com/1424-8220/20/1/38/htmhttps://www.mdpi.com/1424-8220/20/1/38.

Nitin Khanna and Monika Sachdeva. A comprehensive taxonomy of schemes to detect and mitigate blackhole attack and its variants in MANETs. *Computer Science Review*, 32:24–44, 5 2019a. ISSN 1574-0137. .

Nitin Khanna and Monika Sachdeva. Study of trust-based mechanism and its component model in MANET: Current research state, issues, and future recommendation. *International Journal of Communication Systems*, 32(12):e4012, 8 2019b. ISSN 1099-1131. . URL https://onlinelibrary.wiley.com/doi/full/10.1002/dac.4012.

Jln Khatib Sulaiman Dalam No, Sulaiman Muhammed Sulaiman, and Adnan Mohsin Abdulazeez. Leveraging of Gradient Boosting Algorithm in Misuse Intrusion Detection using KDD Cup 99 Dataset. *Indonesian Journal of Computer Science*, 13(1):360, 2 2024. ISSN 2549-7286. . URL http://3.8.6.95/ijcs/index.php/ijcs/article/view/3720.

Joydeep Kundu, Sahabul Alam, and Arindam Dey. Fuzzy based trusted malicious unmanned aerial vehicle detection using in flying ad-hoc network. *Alexandria Engineering Journal*, 99:232–241, 7 2024. ISSN 1110-0168. . URL https://www.sciencedirect.com/science/article/pii/S1110016824004484.

E Edwin Lawrence and R Latha. A Game Theory Approach to Enhance Routing and Security in MANET. *Journal of Network Security*, pages 1–8, 2016. URL www.stmjournals.com.

Alexey V. Leonov and George A. Litvinov. About Applying AODV and OLSR Routing Protocols to Relaying Network Scenario in FANET with Mini-UAVs. *2018 14th International Scientific-Technical Conference on Actual Problems of Electronic Instrument Engineering, APEIE 2018 - Proceedings*, pages 220–228, 11 2018. .

Nathaniel M. Levine and Billie F. Spencer. Post-Earthquake Building Evaluation Using UAVs: A BIM-Based Digital Twin Framework. *Sensors 2022, Vol. 22, Page 873*, 22(3):873, 1 2022. ISSN 1424-8220. . URL https://www.mdpi.com/1424-8220/22/3/873/htmhttps://www.mdpi.com/1424-8220/22/3/873.

Huaizhi Li and Mukesh Singhal. Trust management in distributed systems. *Computer, IEEE*, 40(2):45–53, 2 2007. ISSN 00189162. .

Jun Li, Yifeng Zhou, and Louise Lamont. Communication architectures and protocols for networking unmanned aerial vehicles. *2013 IEEE Globecom Workshops, GC Wkshps 2013*, pages 1415–1420, 2013. .

Yaqun Liu, Jun Xie, Changyou Xing, Shengxu Xie, and Baoan Ni. Construction of FANETs for user coverage and information transmission in disaster rescue scenarios. *Computer Communications*, 207:164–174, 7 2023. ISSN 0140-3664. .

Junwei Lv, Xuegang Hu, Lei Li, and Peipei Li. An effective confidence-based early classification of time series. *IEEE Access*, 7:96113–96124, 2019. ISSN 21693536. .

Mingyang Lyu, Yibo Zhao, Chao Huang, and Hailong Huang. Unmanned Aerial Vehicles for Search and Rescue: A Survey. *Remote Sensing 2023, Vol. 15, Page 3266*, 15 (13):3266, 6 2023. ISSN 2072-4292. . URL https://www.mdpi.com/2072-4292/15/13/3266/htmhttps://www.mdpi.com/2072-4292/15/13/3266.

Bharat Reddy Maddireddy and Bhargava Reddy Maddireddy. Proactive Cyber Defense: Utilizing AI for Early Threat Detection and Risk Assessment. *International Journal of Advanced Engineering Technologies and Innovations*, 1(2):64–83, 12 2020. URL https://ijaeti.com/index.php/Journal/article/view/321.

Sergio Marti, T. J. Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*, pages 255–265. ACM, 2000. . URL https://dl.acm.org/doi/10.1145/345910.345955.

Topside E Mathonsi, Thabiso N Khosa, and Deon P Du Plessis. A Model to Prevent Gray Hole Attack in Mobile Ad-Hoc Networks. *Journal of Advances in Information Technology*, 14(3), 2023. . URL https://www.researchgate.net/publication/371350061.

Jean-aimé Maxa, Slim Ben Mahmoud Mohamed, and Nicolas Larrieu. Survey on UAANET Routing Protocols and Network Security Challenges. *Adhoc & Sensor Wireless Networks*, 37(1-4):231–320, 2017.

Vicente Mayor, Rafael Estepa, Antonio Estepa, and German Madinabeitia. Deploying a Reliable UAV-Aided Communication Service in Disaster Areas. *Wireless Communications and Mobile Computing*, 2019, 2019. ISSN 15308677. .

Rahma Meddeb, Farah Jemili, Bayrem Triki, and Ouajdi Korbaa. A deep learning-based intrusion detection approach for mobile Ad-hoc network. *Soft Computing*, 27(14): 9425–9439, 7 2023. ISSN 14337479. . URL https://link.springer.com/article/10.1007/s00500-023-08324-4.

Salma Badawi Mohammed Ahmed, Syed Aamer Hussain, Liza Abdul Latiff, Norulhusna Ahmad, and Suriani Mohd Sam. Performance Evaluation of FANET Routing Protocols in Disaster Scenarios. *Proceedings of the 2021 IEEE Symposium on Future Telecommunication Technologies, SOFTT 2021*, pages 46–51, 2021. .

Usue Mori, Alexander Mendiburu, Sanjoy Dasgupta, and Jose A. Lozano. Early Classification of Time Series by Simultaneously Optimizing the Accuracy and Earliness. *IEEE Transactions on Neural Networks and Learning Systems*, 29(10):4569–4578, 10 2018. ISSN 21622388. .

Saswati Mukherjee, Matangini Chattopadhyay, Samiran Chattopadhyay, and Pragma Kar. EAER-AODV: Enhanced trust model based on average encounter rate for secure routing in MANET. *Advances in Intelligent Systems and Computing*, 667:135–151, 2018. ISSN 21945357. . URL https://link.springer.com/chapter/10.1007/978-981-10-8183-5_9.

Anand Nayyar. Flying Adhoc Network (FANETs): Simulation Based Performance Comparison of Routing Protocols: AODV, DSDV, DSR, OLSR, AOMDV and HWMP. *2018 International Conference on Advances in Big Data, Computing and Data Communication Systems, icABCD 2018*, 9 2018. .

Shayegan Omidshafiei, Christos Papadimitriou, Georgios Piliouras, Karl Tuyls, Mark Rowland, Jean Baptiste Lespiau, Wojciech M. Czarnecki, Marc Lanctot, Julien Perolat, and Remi Munos. α-Rank: Multi-Agent Evaluation by Evolution. *Scientific Reports*, 9 (1):1–29, 12 2019. ISSN 20452322. . URL https://www.nature.com/articles/s41598-019-45619-9.

Faezeh Pasandideh, João Paulo J. da Costa, Rafael Kunst, Nahina Islam, Wibowo Hardjawana, and Edison Pignaton de Freitas. A Review of Flying Ad Hoc Networks: Key Characteristics, Applications, and Wireless Technologies. *Remote Sensing 2022, Vol. 14, Page 4459*, 14(18):4459, 9 2022. ISSN 2072-4292. . URL https://www.mdpi.com/2072-4292/14/18/4459/htmhttps://www.mdpi.com/2072-4292/14/18/4459.

C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing, 7 2003. URL https://www.rfc-editor.org/info/rfc3561.

Charles Perkins. *Ad Hoc Networking*. Pearson Education India, 2008.

Jon Postel. RFC 768 - User Datagram Protocol, 1980. URL https://www.ietf.org/rfc/rfc768.txt.

Jon Postel. RFC 793 - Transmission Control Protocol, 1981. URL https://www.ietf.org/rfc/rfc793.txt.

Achintya Prakash and Michael P. Wellman. Empirical game-theoretic analysis for moving target defense. *MTD 2015 - Proceedings of the 2nd ACM Workshop on Moving Target Defense, co-located with: CCS 2015*, pages 57–65, 10 2015. . URL https://dl.acm.org/doi/10.1145/2808475.2808483.

P. Rahul and B. Kaarthick. Proficient link state routing in mobile ad hoc network-based deep Q-learning network optimized with chaotic bat swarm optimization algorithm. *International Journal of Communication Systems*, 36(1), 1 2023. ISSN 10991131. .

Dinesh Ramphull, Avinash Mungur, Sheeba Armoogum, and Sameerchand Pudaruth. A review of mobile Ad hoc NETwork (MANET) protocols and their applications. *Proceedings - 5th International Conference on Intelligent Computing and Control Systems, ICICCS 2021*, pages 204–211, 5 2021. .

Pooja Rani, Kavita, Sahil Verma, and Gia Nhu Nguyen. Mitigation of Black Hole and Gray Hole Attack Using Swarm Inspired Algorithm with Artificial Neural Network. *IEEE Access*, 8:121755–121764, 2020. ISSN 21693536. .

Abderahman Rejeb, Karim Rejeb, Steve Simske, and Horst Treiblmaier. Humanitarian Drones: A Review and Research Agenda. *Internet of Things*, 16:100434, 12 2021. ISSN 2542-6605. .

Joonsu Ryu and Sungwook Kim. Reputation-Based Opportunistic Routing Protocol Using Q-Learning for MANET Attacked by Malicious Nodes. *IEEE Access*, 11: 47701–47711, 2023. ISSN 21693536. .

Geetha Sadayan and Karthiyayini Ramaiah. Enhanced data security in MANET using trust-based Bayesian statistical model with RSSI by AOMDV. *Concurrency and Computation: Practice and Experience*, 34(8):e5397, 4 2022. ISSN 1532-0634. . URL https://onlinelibrary.wiley.com/doi/full/10.1002/cpe.5397.

Leovigildo Sánchez-Casado, Gabriel Maciá-Fernández, Pedro García-Teodoro, and Roberto Magán-Carrión. A model of data forwarding in MANETs for lightweight detection of malicious packet dropping. *Computer Networks*, 87:44–58, 7 2015. ISSN 1389-1286. .

L. J. Savage. The Theory of Statistical Decision. *Journal of the American Statistical Association*, 46(253):55, 3 1951. ISSN 01621459. .

Oussama Sbai and Mohamed Elboukhari. Mobile Ad Hoc networks intrusion detection system against packet dropping attacks. *Indonesian Journal of Electrical Engineering and Computer Science*, 26(2):819–825, 2022. ISSN 2502-4752. .

Hichem Sedjelmaci, Sidi Mohammed Senouci, and Nirwan Ansari. A Hierarchical Detection and Response System to Enhance Security Against Lethal Cyber-Attacks in UAV Networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(9): 1594–1606, 9 2018. ISSN 21682232. .

Hichem Sedjelmaci, Imane Horiya Brahmi, Nirwan Ansari, and Mubashir Husain Rehmani. Cyber Security Framework for Vehicular Network Based on a Hierarchical Game. *IEEE Transactions on Emerging Topics in Computing*, 9(1):429–440, 1 2021. ISSN 21686750. .

Sevil Sen and John A. Clark. Evolutionary computation techniques for intrusion detection in mobile ad hoc networks. *Computer Networks*, 55(15):3441–3457, 10 2011. ISSN 1389-1286. .

Antesar M. Shabut, Keshav P. Dahal, Sanat Kumar Bista, and Irfan U. Awan. Recommendation Based Trust Model with an Effective Defence Scheme for MANETs. *IEEE Transactions on Mobile Computing*, 14(10):2101–2115, 10 2015. ISSN 15361233. .

Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *International Conference on Information Systems Security and Privacy*, 2018. ISBN 9789897582820. .

Anshul Sharma and Sanjay Kumar Singh. A novel approach for early malware detection. *Transactions on Emerging Telecommunications Technologies*, 32(2), 2 2021. ISSN 21613915. .

Anshul Sharma, Abhinav Kumar, and Sanjay Kumar Singh. Early classification of time series data: overview, challenges, and opportunities. *Data Fusion Techniques and Applications for Smart Healthcare*, pages 227–250, 1 2024. .

Devu Manikantan Shila and Tricha Anjali. A game theoretic approach to gray hole attacks in wireless mesh networks. In *Proceedings - IEEE Military Communications Conference MILCOM*, 2008. ISBN 9781424426775. .

Mario Silvagni, Andrea Tonoli, Enrico Zenerino, and Marcello Chiaberge. Multipurpose UAV for search and rescue operations in mountain avalanche events. *Geomatics, Natural Hazards and Risk*, 2016. ISSN 1947-5713. . URL https://www.tandfonline.com/action/journalInformation?journalCode=tgnh20.

Gurpreet Singh. Comparison of Wi-Fi IEEE 802.11 Standards Relating to Media Access Control Protocols. *Article in International Journal of Computer Science and Information Security*, 2016. URL https://sites.google.com/site/ijcsis/.

Kuldeep Singh and Anil Kumar Verma. TBCS: A Trust Based Clustering Scheme for Secure Communication in Flying Ad-Hoc Networks. *Wireless Personal Communications*, 114(4):3173–3196, 10 2020. ISSN 1572834X. . URL https://link.springer.com/article/10.1007/s11277-020-07523-8.

Sandeep Singh, Anshu Bhasin, and Anshul Kalia. Capitulation of mitigation techniques of packet drop attacks in MANET to foreground nuances and ascertain trends.

International Journal of Communication Systems, 34(10):e4822, 7 2021. ISSN 1099-1131. . URL https://onlinelibrary.wiley.com/doi/full/10.1002/dac.4822https://onlinelibrary.wiley.com/doi/abs/10.1002/dac.4822https://onlinelibrary.wiley.com/doi/10.1002/dac.4822.

Sheevendra Singh, Isha Sharma, Praneet Saurabh, and Ritu Prasad. Fuzzy Logic Based Packet Dropping Detection Approach for Mobile Ad-Hoc Wireless Network. *Advances in Intelligent Systems and Computing*, 1057:263–273, 2020. ISSN 21945365. URL https://link.springer.com/chapter/10.1007/978-981-15-0184-5{_}23.

Skyfront. Perimeter 8 UAS Specifications. *https://skyfront.com/perimeter-8*, 2024. URL https://skyfront.com/perimeter-8.

Ashish Srivastava and Jay Prakash. Future FANET with application and enabling techniques: Anatomization and sustainability issues. *Computer Science Review*, 39: 100359, 2 2021. ISSN 1574-0137. .

Fan Wei Lee Wenke Prodromidis Andreas Stolfo Salvatore and Philip Chan. KDD Cup 1999 Data. UCI Machine Learning Repository, 1999.

Basant Subba, Santosh Biswas, and Sushanta Karmakar. A game theory based multi layered intrusion detection framework for VANET. *Future Generation Computer Systems*, 82:12–28, 5 2018. ISSN 0167-739X. .

Richard S. Sutton and Andrew G. Barto. *Reinforcement learning : an introduction*. MIT Press, 2018. ISBN 9780262352703.

Andrew Tanenbaum, David Wetherall, and Nick Feamster. *Computer Networks*. TGJones, 2021. ISBN 9781292374062. URL https://www.tgjonesonline.co.uk/Product/Andrew-Tanenbaum/Computer-Networks-Global-Edition/11788666.

Dan Tang, Xiyin Wang, Xiong Li, Pandi Vijayakumar, and Neeraj Kumar. AKN-FGD: Adaptive Kohonen Network Based Fine-Grained Detection of LDoS Attacks. *IEEE Transactions on Dependable and Secure Computing*, 20(1):273–287, 1 2023. ISSN 19410018. .

Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the KDD CUP 99 data set. *IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009*, 12 2009. .

The Washington Post. Elon Musk's control over satellite internet demands a reckoning, 2023. URL https://www.washingtonpost.com/opinions/2023/08/30/elon-musk-starlink-internet-satellite-ukraine/.

Mohammed Hussein Thwaini. Anomaly Detection in Network Traffic using Machine Learning for Early Threat Detection. *Data and Metadata*, 1:34–34, 12 2022. ISSN 2953-4917. . URL

https://dm.saludcyt.ar/index.php/dm/article/view/72/183https://dm.saludcyt.ar/index.php/dm/article/view/72.

Kai Ming Ting. Precision and Recall. *Encyclopedia of Machine Learning*, pages 781–781, 2011. . URL https://link.springer.com/referenceworkentry/10.1007/978-0-387-30164-8_652.

Tom Henderson, George Riley, Sally Floyd, and Sumit Roy. ns-3 — a discrete-event network simulator for internet systems, 2008. URL https://www.nsnam.org/.

Mauro Tropea, Mattia Giovanni Spina, Abderrahmane Lakas, Panagiotis Sarigiannidis, and Floriano De Rango. SecGPSR: A secure GPSR protocol for FANET against Sybil and Gray Hole Attacks. *IEEE Access*, 2024. ISSN 21693536. .

Kai Yun Tsao, Thomas Girdler, and Vassilios G. Vassilakis. A survey of cyber security threats and solutions for UAV communications and flying ad-hoc networks. *Ad Hoc Networks*, 133:102894, 8 2022. ISSN 1570-8705. .

Kefayat Ullah and Prodipto Das. Trust-Based Routing for Mitigating Grayhole Attack in MANET. *Proceedings of the International Conference on Computing and Communication Systems*, 24:713–721, 2018. ISSN 23673389. . URL https://link.springer.com/chapter/10.1007/978-981-10-6890-4_68.

V. Uma Rani, J. Jebamalar Tamilselvi, Rajan John, and J. Deepa. Trust Model for Secure Routing in Wireless Sensor Network using AI Technique. *8th International Conference on Smart Structures and Systems, ICSSS 2022*, 2022. .

Vijay Varadharajan and Venkatesan Balakrishnan. PACKET DROP ATTACK: A SERIOUS THREAT TO OPERATIONAL MOBILE AD HOC NETWORKS. In *Proceedings of the international conference on networks and communication systems*, pages 89–95, Krabi, 2005. URL https://www.researchgate.net/publication/266455657.

Ashish Vaswani, Google Brain, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. *Advances in Neural Information Processing Systems*, 30, 2017.

Styliani Verykokou, Anastasios Doulamis, George Athanasiou, Charalabos Ioannidis, and Angelos Amditis. UAV-based 3D modelling of disaster scenes for Urban Search and Rescue. *IST 2016 - 2016 IEEE International Conference on Imaging Systems and Techniques, Proceedings*, pages 106–111, 11 2016. .

Amol D. Vibhute, Chandrashekhar H. Patil, Arjun V. Mane, and Karbhari V. Kale. Towards Detection of Network Anomalies using Machine Learning Algorithms on the NSL-KDD Benchmark Datasets. *Procedia Computer Science*, 233:960–969, 1 2024. ISSN 1877-0509. .

S Vijayalakshmi, S Bose, G Logeswari, and T Anitha. Hybrid defense mechanism against malicious packet dropping attack for MANET using game theory. *Cyber Security and Applications*, 1:100011, 12 2023. ISSN 2772-9184. . URL https://linkinghub.elsevier.com/retrieve/pii/S277291842200011X.

Christopher Watkins. *Learning From Delayed Rewards*. PhD thesis, 1989.

Zhexiong Wei, Helen Tang, F. Richard Yu, Maoyu Wang, and Peter Mason. Security enhancements for mobile ad hoc networks with trust management using uncertain reasoning. *IEEE Transactions on Vehicular Technology*, 63(9):4647–4658, 11 2014. ISSN 00189545. .

Gary M. Weiss and Foster Provost. Learning When Training Data are Costly: The Effect of Class Distribution on Tree Induction. *Journal of Artificial Intelligence Research*, 19: 315–354, 10 2003. ISSN 1076-9757. . URL https://www.jair.org/index.php/jair/article/view/10346.

Michael P Wellman. Methods for Empirical Game-Theoretic Analysis (Extended Abstract). *AAAI'06: proceedings of the 21st national conference on Artificial intelligence*, 2: 1552–1555, 7 2006.

Michael P. Wellman and Achintya Prakash. Empirical game-theoretic analysis of an adaptive cyber-defense scenario (preliminary report). *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8840:43–58, 2014. ISSN 16113349. . URL https://link.springer.com/chapter/10.1007/978-3-319-12601-2_3.

Michael P. Wellman, Thanh H. Nguyen, and Mason Wright. Empirical game-theoretic methods for adaptive cyber-defense. *Lecture Notes in Computer Science*, 11830 LNCS: 112–128, 2019. ISSN 16113349. . URL https://link.springer.com/chapter/10.1007/978-3-030-30719-6_6.

Michael P. Wellman, Karl Tuyls, and Amy Greenwald. Empirical Game Theoretic Analysis: A Survey. *Journal of Artificial Intelligence Research*, 82:1017–1076, 2 2025. ISSN 1076-9757. . URL https://www.jair.org/index.php/jair/article/view/16146.

Ali H. Wheeb, Rosdiadee Nordin, Asma' Abu Samah, and Dimitris Kanellopoulos. Performance Evaluation of Standard and Modified OLSR Protocols for Uncoordinated UAV Ad-Hoc Networks in Search and Rescue Environments. *Electronics 2023, Vol. 12, Page 1334*, 12(6):1334, 3 2023. ISSN 2079-9292. . URL https://www.mdpi.com/2079-9292/12/6/1334/htmhttps://www.mdpi.com/2079-9292/12/6/1334.

Jason Whelan, Thanigajan Sangarapillai, Omar Minawi, Abdulaziz Almehmadi, and Khalil El-Khatib. Novelty-based Intrusion Detection of Sensor Attacks on Unmanned Aerial Vehicles. *Q2SWinet 2020 - Proceedings of the 16th ACM Symposium on QoS and*

Security for Wireless and Mobile Networks, pages 23–28, 11 2020. . URL https://dl.acm.org/doi/10.1145/3416013.3426446.

Bryce Wiedenbeck, Ben-Alexander Cassell, and Michael P Wellman. Bootstrap Statistics for Empirical Games. In *AAMAS*, 2014. URL www.ifaamas.org.

Lai Leng Woo, Mark Zwolinski, and Basel Halak. Early detection of system-level anomalous behaviour using hardware performance counters. *Proceedings of the 2018 Design, Automation and Test in Europe Conference and Exhibition, DATE 2018*, 2018-January:485–490, 4 2018. .

Celimuge Wu, Satoshi Ohzahata, and Toshihiko Kato. Flexible, Portable, and Practicable Solution for Routing in VANETs: A Fuzzy Constraint Q-Learning Approach. *IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY*, 62(9):4251, 2013. .

Bo Yang, Ryo Yamamoto, and Yoshiaki Tanaka. Dempster-Shafer evidence theory based trust management strategy against cooperative black hole attacks and gray hole attacks in MANETs. *International Conference on Advanced Communication Technology, ICACT*, pages 223–232, 2014. ISSN 17389445. .

Inoue Yuki. Earthquake in Japan: Drone technology helps disaster relief efforts, 2024. URL https://www3.nhk.or.jp/nhkworld/en/news/backstories/3406/.

Wenbin Zhai, Liang Liu, Youwei Ding, Shanshan Sun, and Ying Gu. ETD: An Efficient Time Delay Attack Detection Framework for UAV Networks. *IEEE Transactions on Information Forensics and Security*, 18:2913–2928, 2023. ISSN 15566021. .

Min Zhang, Chao Dong, Peng Yang, Ting Tao, Qihui Wu, and Tony Q.S. Quek. Adaptive Routing Design for Flying Ad Hoc Networks. *IEEE Communications Letters*, 26(6):1438–1442, 6 2022. ISSN 15582558. .

Ziming Zhao, Hongxin Hu, Gail Joon Ahn, and Ruoyu Wu. Risk-aware mitigation for MANET routing attacks. *IEEE Transactions on Dependable and Secure Computing*, 9(2): 250–260, 2012. ISSN 15455971. .

Alice Zheng and Amanda Casari. *Feature engineering for machine learning: principles and techniques for data scientists*. " O'Reilly Media, Inc.", 2018.