2024 IEEE Intelligent Vehicle Symposium (IV) | 979-8-3503-4881-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/IV55156.2024.10588845

DRIFT: Resilient Distributed Coordinated Fleet Management Against Communication Attacks

Richard Owoputi[1], Srivalli Boddupalli[2], Jabari Wilson[3], and Sandip Ray[1]
[1]Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA.
rowoputi@ufl.edu, sandip@ece.ufl.edu
[2]Lucid Motors, Newark, CA 94560, USA. sv.boddupalli@gmail.com
[3]Department of Engineering Education, University of Florida, Gainesville, FL 32611, USA.
jabari.wilson@ufl.edu

Abstract— **Consider a fleet of autonomous vehicles traversing an adversarial terrain that includes obstacles and mines. The goal of the fleet is to ensure that they can complete their mission safely (with minimal casualty) and efficiently (as quickly as possible). In Distributed Coordinated Fleet Management (DCFM), fleet members coordinate with one another while traversing the terrain, *e.g.*, a vehicle encountering an obstacle at a location l can inform other agents so that they can recompute their route to avoid l. In this paper, we consider the problem of *cyber-resilient DCFM*, *i.e.*, DCFM in an environment where the adversary can additionally tamper with the cyber-communication performed by the fleet members. Our framework, DRIFT, enables fleet members to coordinate in the presence of such adversaries. Our extensive evaluations demonstrate that DRIFT can achieve a high degree of safety and efficiency against a large spectrum of communication adversaries.**

I. INTRODUCTION

Consider a fleet of autonomous vehicles traversing an adversarial terrain that includes obstacles and mines planted by an adversary. The goal of the fleet is to ensure that they can complete the mission (*i.e.*, complete their travel through this terrain) as quickly as possible and with minimal casualty. Since each vehicle has a limited field of view and limited prior knowledge of the region, one way for vehicles to achieve this is to exchange messages informing each other of the state of the terrain in their neighborhood. For instance, a vehicle encountering an obstacle at a particular terrain location can alert the other vehicles to determine an alternate route. A protocol to achieve this coordination will be referred to *Distributed Coordinated Fleet Management* (DCFM) in this paper. DCFM is a common requirement in various domains, such as military navigation and geographical expedition [1]–[3].

We study DCFM in the context of potential cyber-attacks by an adversary that can corrupt the communication from the vehicles. An adversary's goal is to mislead fleet members into making detrimental decisions. For instance, if a vehicle $\mathcal{S}$ falsely claims a mine at location $\mathcal{L}$, another vehicle $\mathcal{V}$ might reroute, reducing efficiency. Conversely, ignoring a real threat at $\mathcal{L}$ based on $\mathcal{S}$ report can lead to casualties.

Cyber-resilient DCFM aims to ensure mission success with minimal safety and efficiency loss despite such adversaries which is a critical requirement in the adoption of DCFM for military applications.

This paper proposes a method, DRIFT (Distributed Resilient Fleet), to bolster DCFM against cyber threats. It suggests that a vehicle $\mathcal{V}$, upon receiving data about $\mathcal{L}$ from $\mathcal{S}$, should not fully trust it but consider previous knowledge about $\mathcal{L}$ and the trustworthiness of $\mathcal{S}$. We define confidence and trust in this context and show how DRIFT can effectively ensure cyber-resilience, as evidenced by extensive evaluations in a grid-based routing scenario with real-world challenges like mines and obstacles.

The remainder of the paper is organized as follows. Section II discusses related research in security of distributed wireless and cyber-physical systems under communication attacks. We formally present the DCFM in Section III and cyber-resilient DFCM in Section IV. The DRIFT solution is discussed in Section V. We present an empirical evaluation of DRIFT in Section VI and conclude in Section VII.

II. RELATED WORK

Numerous studies have been dedicated to enhancing the security of wireless infrastructures against cyber threats. Cryptographic [4] and Blockchain-enabled frameworks [5] are proposed for message security. Authentication techniques validate vehicle identities [6]–[9], and honeypots alongside intrusion detection monitor attacker behaviors [10]–[14]. While cryptographic and authentication techniques guard against many vehicular threats, they are less effective against communication attacks. In particular, responses to communication attacks need to be swift and dynamic to be effective in the context of a fast-moving vehicle. This makes it difficult to employ approaches that require intensive, onboard computation.

Another line of related research exploits game-theoretic models to develop quantitative solutions to different adversarial scenarios [15]–[17]. A critical problem with these approaches is the need to model the games to reflect the complete information and inference available to the different players [18]. Furthermore, many game-theoretic solutions rely on strong assumptions on the rationality of the adversary.

Additionally, trust-based routing protocols [19] [20] enhance vehicular network security with lower time overhead compared to cryptographic solutions. These methods focus on point-to-point communications and may not cover multicast scenarios where an adversary impacts a sender and multiple receivers.

III. DCFM Formulation

A. Convention and Preliminaries

We formalize the terrain being traversed by the fleet as an undirected graph $\mathcal{M}_G = \langle L, E\rangle$, where L is a set of *location nodes*. For each node $l \in L$, we use $\textsf{\textit{nbrs}}(l)$ to refer to the set of nodes that share an edge with l in M. We refer to $\mathcal{M}$ as the *global map*. Each edge e in $\mathcal{M}_G$ between location nodes l and l', where $l' \in \textsf{\textit{nbrs}}(l)$, is labeled with a nonnegative real number, which represents the *distance* between l and l'. Finally, each location l is assumed to be associated with two components: (1) a *unique identifier*, denoted by l[ID];[1] and (2) an attribute from the set $\{\text{NORMAL}, \text{OBSTACLE}, \text{MINE}\}$, which we refer as *terrain type* of l and denote by l[TYPE].

We assume that the fleet members have a notion of the terrain being traversed, but not a completely accurate notion of the location type. To formalize this, each vehicle $\mathcal{V}$ includes "local instance" of the map, which we call the *local map* of $\mathcal{V}$ denoted by $\mathcal{V}[\text{MAP}]$. We assume that the local copies are *consistent* in the following sense. Suppose $\mathcal{V}$ is a member of a fleet traversing a terrain defined by the map $\mathcal{M}_G$. Then (1) for every pair of locations $l \in \mathcal{M}_G$ there exists a location $l_\mathcal{V} \in \mathcal{V}[\text{MAP}]$ and vice versa; and (2) for any pair of locations $l, l' \in \mathcal{M}$ there are locations $l_\mathcal{V}, l'_\mathcal{V} \in \mathcal{V}[\text{MAP}]$ such that there is an edge between $l_\mathcal{V}$ and $l'_\mathcal{V}$ if and only if there is an edge between l and l' in $\mathcal{M}_G$ and the distance between $l_\mathcal{V}$ and $l'_\mathcal{V}$ in $\mathcal{V}[\text{MAP}]$ is the same as the distance between l and l' in $\mathcal{M}_G$. Informally, the consistency requirement implies the vehicle's understanding of the geographical positions of the points in the terrain matches reality, although their understanding of the terrain type (*i.e.*, whether the type is NORMAL, OBSTACLE, or MINE) may not match. Based on this notation, we will feel free to use the same symbols l and l' to either denote the locations in global map $\mathcal{M}_G$ or the local map $\mathcal{V}[\text{MAP}]$ when the intended map is clear from context. Furthermore, when we talk about the location l in $\mathcal{M}_G$ and $\mathcal{V}[\text{MAP}]$ we mean l in $\mathcal{M}_G$ and $l_\mathcal{V}$ in $\mathcal{V}[\text{MAP}]$.

B. Fleet Management and Coordination

The goal of fleet management is for each member $\mathcal{V}$ of the fleet to find an "optimal" route from its current location to a designated *destination location* $\mathcal{D} \in \mathcal{M}$ while avoiding obstacles and mines. Formally, given a a map $\mathcal{M}$ *start location* S and a *destination location* D, a *route* R from S to D in map $\mathcal{M}$ is a sequence of nodes $R \doteq \langle l_1, \ldots, l_k\rangle$ such that (1) $l_1 = S$, (2) $l_k = D$, and (3) for each $i = 1, \ldots, k-1$ there is an edge in $\mathcal{M}$ between l_i and l_{i+1}.

TABLE I
Parameter Definitions

Parameter	Definition
$\mathcal{V}$	Ego Vehicle
$\mathcal{S}$	Sender
$\mathcal{M}_G$	Global Map
$l_\mathcal{V}$	Current location of Ego Vehicle $\mathcal{V}$ in $\mathcal{M}_G$
$l_\mathcal{S}$	Current location of Sender in $\mathcal{M}_G$
RoI	2-D Region of interest
λ	Confidence 3-tuple of current location
T_S	Trust assigned to S by $\mathcal{V}$
ΔT	Trust offset (to increment or decrement T_S)
Dissimilarity Index	Weighted difference between $\lambda\mathcal{V}$ and $\lambda\int$
Similarity Index	
T_{max}	Maximum trust index (Constant)
SF	Normalizes the product of T_S and T_{max} over their squared sum

Definition 1 (Viability). *Let $\mathcal{M}$ be a map, $R \doteq \langle l_1, \ldots, l_k\rangle$ be a route from S to D in $\mathcal{M}$. R will be called* viable *in $\mathcal{M}$ if for each $i = 1, \ldots, k$, $l_i[TYPE] =$* NORMAL.

Note that the consistency requirement above implies that a route R in the global map $\mathcal{M}_G$ is also a route in the local map of every vehicle $\mathcal{V}$. However, the viability of a route can vary, *e.g.*, a route R that is viable in the global map $\mathcal{M}_G$ may not be viable in $\mathcal{V}[\text{MAP}]$ for some member $\mathcal{V}$ of the fleet.

The idea of DCFM is for vehicles to coordinate movement by periodically broadcasting the terrain type of their current location. Algorithm 1 provides a high-level overview of DCFM functionality for vehicle $\mathcal{V}$. Note that $\mathcal{V}$ broadcasts [2] a message whenever the terrain type of the current location is not NORMAL (Line 15) or if a certain time interval has elapsed from its previous broadcast (Line 13). The message structure (Line 12) encapsulates the location l in the global map $\mathcal{M}_G$, which represents the current location of the Ego vehicle $\mathcal{V}$, terrain type l[TYPE], and a flag signifying the availability of an optimal route $\mathcal{R}_{opt}$.

Remark. *The algorithm* UpdateRoute *determines the shortest route $\mathcal{R}_{opt}$ from the current location $l_\mathcal{V}$ of $\mathcal{V}$ in the subgraph $\mathcal{M}_{\textbf{ROI}}$ of $\mathcal{M}$. We refer to the subgraph as the* current region of interest. *The entire map is not used for computing the R_{opt} to achieve computational efficiency. Instead, a vehicle searches for a route within the region of interest, and if no such route is found, then the diameter of the region of interest is progressively increased to explore larger areas of the map.*

[3]

[1] Informally, the identifier of a location is given by its position in the map of the terrain, *e.g.*, in practice it can be specified by the latitude and longitude.

[2] Our communication model assumes that any message broadcast by a sender vehicle is received only by vehicles located within a certain distance from $\mathcal{S}$ at the time of transmission. We refer to this distance as *broadcast range*, and the value of the broadcast range is a parameter to the problem. The DCFM algorithm (and subsequently DRIFT) does not use this parameter in the computation. However, it is used during the evaluation of the algorithms in Section VI.

[3] The scaling factor SF determines the weight of the trust values (T_s and T_{max}) when updating the location, computed as the product of current trust T_s and the maximum trust T_{max}, normalized by the sum of the square of T_s and T_{max}.

Algorithm 1 DCFM

```
 1: procedure DCFM(Ego: $\mathcal{V}$; Sender: $S$)
 2:     $InitializeMap\mathcal{M}()$
 3:     $InitializeRoute\mathcal{R}_{init}$
 4:     $cur\_time \leftarrow 0$
 5:     $next\_broadcast\_time \leftarrow cur\_time + T_B$
 6:     while $\mathcal{V}$ not reached destination do
 7:         for $msg$ in $in\_buf$ do
 8:             $l_\mathcal{S}, l[\text{TYPE}]^S \leftarrow$ ReadMsg$(msg)$
 9:             $\mathcal{V}[\text{MAP}] \leftarrow$ UpdateMap$(l_\mathcal{V}, l_\mathcal{S},$
                        $\mathcal{M}, l[\text{TYPE}]^S)$
10:         $\mathcal{R}_{opt}, flag \leftarrow$ UpdateRoute$(l_\mathcal{V}, l_{des},$
                        $\mathcal{V}[\text{MAP}], \mathcal{R}_{init})$
11:         $\mathcal{R}_{init} \leftarrow \mathcal{R}_{opt}$
12:         $msg \leftarrow$ CreateMsg$(l_\mathcal{V}, \{l[\text{TYPE}]^\mathcal{V}, flag)$
13:         if $cur\_time$ is $next\_broadcast\_time$ then
14:             Broadcast$(msg)$
15:         else if $l[\text{TYPE}]^\mathcal{V}$ is not NORMAL then
16:             Broadcast$(msg)$
17:         $next\_broadcast\_time \leftarrow cur\_time + T_B$
18:         if $flag \leftarrow$ False then
19:             $(l_{v(next)}) \leftarrow$ GetNextLoc$(l_\mathcal{V}, \mathcal{R}_{opt})$
20:             MoveTo$(l_{v(next)})$
21:             $cur\_time \leftarrow cur\_time + 1$
22:         else if $flag \leftarrow$ True then
23:             VehicleFailure
24:
25: procedure UPDATEROUTE($l$, $l_{des}$, $\mathcal{M}$, $\mathcal{R}$)
26:     $InitializeRadius\mathcal{R}, \mathcal{R}_{max}$
27:     $\Delta\mathcal{R} \leftarrow 1$
28:     for $n$ in $\mathcal{R}$ do
29:         $\mathcal{M}_{RoI} \leftarrow$ nodes in $\mathcal{M}$ within radius $\mathcal{R}$ of $n$
30:     $\mathcal{R}_{opt} \leftarrow$ Route from $l$ to $l_{des}$within $\mathcal{M}_{RoI}$
31:     if $\mathcal{R}_{opt}$ exists then
32:         $flag \leftarrow$ False
33:         break
34:     else
35:         $flag \leftarrow$ True
36:         $\mathcal{R} \leftarrow \mathcal{R} + \Delta\mathcal{R}$
37:         while $\mathcal{R} \leq \mathcal{R}_{max}$ do
38:             for $n$ in $\mathcal{R}_{opt}$ do
39:                 $\mathcal{M}_{RoI} \leftarrow$ nodes in $\mathcal{M}$ within $\mathcal{R}$ of $n$
40:             $\mathcal{R}_{opt} \leftarrow$ path from $l$ to $l_{des}$ within $\mathcal{M}_{RoI}$
41:             if $\mathcal{R}_{opt}$ exists or $\mathcal{R} = \mathcal{R}_{max}$ then
42:                 break
43:     return $R_{opt}, flag$
44: procedure UPDATEMAP($l$, $\mathcal{M}$, $l[\text{TYPE}]$)
45:     for $l$ in $\mathcal{M}$ do
46:         $l.l[\text{TYPE}] \leftarrow l[\text{TYPE}]^S$
```

IV. Cybersecurity Challenges in DCFM and DRiFt solution

Algorithm 1 implicitly assumes that the messages received by a vehicle $\mathcal{V}$ are trustworthy; this permits the receiving vehicle $\mathcal{V}$ to adjust its route based on the information received from sender $\mathcal{S}$. An adversary can disrupt the algorithm by sending wrong or misleading messages on behalf of $\mathcal{S}$, either by masquerading as $\mathcal{S}$ or by hacking and compromising the design and implementation of $\mathcal{S}$. The goal of *cyber-resilient fleet management* is for vehicles in the fleet to complete the mission in the presence of such adversaries.

Threat Model

The threat model accounts for the potential presence of rogue vehicles within the fleet. These vehicles are characterized by their capability to transmit incorrect or misleading messages, specifically by altering the terrain type information within the message payload. The formal definition is as follows:

Let $\mathcal{L}$ represent a location in the global map $\mathcal{M}_G$. If $\mathcal{L}'$ is derived from $\mathcal{L}$ by modifying $\mathcal{L}[\text{TYPE}]$ to a different value x, not equal to the original $\mathcal{L}[\text{TYPE}]$, then any message m' containing the payload $\mathcal{L}'$ is considered an *adversarial mutation* of location $\mathcal{L}$. Within this framework, a rogue sender vehicle $\mathcal{S}$ is allowed to diverge from the expected behavior dictated by Algorithm 1 in the following ways:

1) $\mathcal{S}$ may opt to broadcast a message m', which could be an adversarial mutation of its current location l, diverging from the original message formulated in line 12 of the algorithm. Moreover, this rogue vehicle can send the mutated message m' at any arbitrary time, either in addition to or instead of the messages scheduled for broadcast according to the algorithm (specifically, Lines 14 and 16). This flexibility extends to the point where $\mathcal{S}$ might choose not to broadcast any message at a given time, even if such an action contradicts the directives of Algorithm 1.
2) The model does not limit the number of rogue vehicles; however, it assumes that these vehicles act independently without collusion or collaboration.
3) A DoS attack in this context manifest through a rogue vehicle by refraining from sending critical messages. Any vehicle engaging in such disruptive behavior is considered an attacker, as its actions directly impede DCFM's functionality.

Although the primary focus is on rogue sender vehicles, the threat model equivalently addresses the risk of Man in the Middle (MITM) attacks. In such scenarios, the disruptive actions of an MITM adversary interfering with the communication between two vehicles, say $\mathcal{U}$ and $\mathcal{V}$, are modeled by treating $\mathcal{U}$ as if it were a rogue sender. This attack encompasses the interception and potential alteration of messages in transit.

V. Introduction to DRiFt

Our approach to addressing the cybersecurity challenges is a new algorithm, DRiFt, that extends the DCFM algorithm

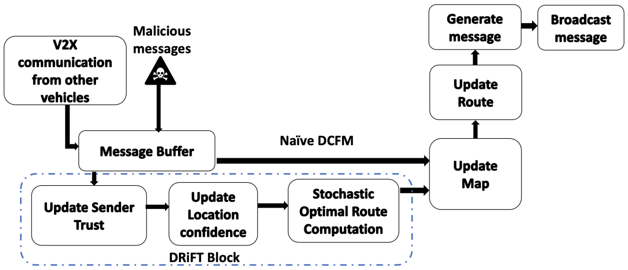

Fig. 1. DCFM and DRIFT architecture

Algorithm 2 DRIFT

```
procedure DRIFT(Ego: $\mathcal{V}$; Sender: $S$)
    $InitializeMap\mathcal{M}()$
    $InitializeRoute\mathcal{R}_{init}$
    $InitializeLocationConfidence\lambda\mathcal{V}$
    $InitializeSenderTrustT_S$
    $cur\_time \leftarrow 0$
    $next\_broadcast\_time \leftarrow cur\_time + T_B$
    while $\mathcal{V}$ not reached destination do
        for $msg$ in $in\_buf$ do
            $l_{\mathcal{S}}, l[\text{TYPE}]^{\mathcal{S}} \leftarrow \textit{ReadMsg}(msg)$
            $\lambda\mathcal{S} \leftarrow \textit{CreateSenderConfidence}(l[\text{TYPE}])^{\mathcal{S}}$
            $T_S \leftarrow \textit{UpdateSenderTrust}(T_S, \lambda\mathcal{V}, \lambda\mathcal{S})$
            $\lambda\mathcal{V} \leftarrow \textit{UpdateLocConfidence}(T_S, \lambda\mathcal{V}, \lambda\mathcal{S})$
            $\mathcal{V}[\text{MAP}] \leftarrow \textit{UpdateMap}(l_{\mathcal{V}}, \mathcal{M},$
                    $l[\text{TYPE}]^{\mathcal{S}}, \lambda\mathcal{V}, T_S, T_{\max}, \lambda\mathcal{S})$
        $\mathcal{R}_{opt}, flag \leftarrow \textit{UpdateRoute}(l_{\mathcal{V}},$
                $l_{des}, \mathcal{V}[\text{MAP}], \mathcal{R}_{init})$
        $\mathcal{R}_{init} \leftarrow \mathcal{R}_{opt}$
        $msg \leftarrow \textit{CreateMsg}(l_{\mathcal{V}}, \{l[\text{TYPE}]^{\mathcal{V}}, flag)$
        if $cur\_time$ is $next\_broadcast\_time$ then
            $\textit{Broadcast}(msg)$
        else if $\{l[\text{TYPE}]^{\mathcal{V}}$ is not $normal$ then
            $\textit{Broadcast}(msg)$
        $next\_broadcast\_time \leftarrow cur\_time + T_B$
        if $flag \leftarrow$ False then
            $(l_{v(next)}) \leftarrow \textit{GetNextLoc}(l_{\mathcal{V}}, \mathcal{R}_{opt})$
            $\textit{MoveTo}(l_{v(next)})$
            $cur\_time \leftarrow cur\_time + 1$
        else if $flag \leftarrow$ True then
            VehicleFailure
```

to enable each vehicle in the fleet to dynamically determine the sender's legitimacy before relying on the received communication. Fig. 1 shows the different functional components of DRIFT. The idea is for each vehicle to first determine the sender's trustworthiness and update its notion of the map accordingly. Each vehicle then determines the most optimal route R_{opt} to the destination based on prior knowledge and the information it received from others.

Algorithm 2 defines the flow of operations in DRIFT. The key idea is for each vehicle $\mathcal{V}$ to maintain the following pieces of information.

- **Location Node Confidence:** λ is a 3-tuple (C_n, C_o, C_m), which represents the estimate of vehicle $\mathcal{V}$ for node L to be a normal node, an obstacle or a mine respectively. The sum $C_n + C_o + C_m = 1$.
- **Sender Trust:** $T_{\mathcal{V}S}$ represents the trust assigned by $\mathcal{V}$ to a sender $\mathcal{S}$. $T_{\mathcal{V}S}$ is a non-zero positive value in the range $[T_{min}, T_{max}]$, where T_{max} is a parameter of the system (see Table I).

We say that $\mathcal{V}$ is in conflict with message m from sender $\mathcal{S}$ if the location type in the payload of m received from $\mathcal{S}$ for location l does not match location type for l that corresponds to the highest confidence in the confidence tuple of $\mathcal{V}$. For instance, if the message from the sender stipulates location l to be NORMAL while the C_m has the largest value in the confidence tuple for l suggesting that $\mathcal{V}$ "believes" that l has location type MINE, then we will consider $\mathcal{V}$ to be in conflict with m. In case of conflict, vehicle $\mathcal{V}$ adjusts the confidence tuple to account for the information but also adjusts the value of its trust of $\mathcal{S}$ to account for the conflict. We introduce an additional parameter SF, which is a scaling factor that determines the weight of the trust values when updating the location $\lambda\mathcal{V}'$. Algorithm 3 explains these updates. They work as follows.

a) Sender Trust Computation: $\mathcal{V}$ updates the sender's trust based on the degree of dissimilarity between what it knows prior and what the sender reports. If the dissimilarity is low, $\mathcal{V}$ trusts the sender and increments the trust assigned to it. If dissimilarity is high, $\mathcal{V}$ is more skeptical about the sender and decrements the trust assigned to it. Algorithm 3 shows the equation governing the sender trust update step.

b) Location Confidence Computation: $\mathcal{V}$ updates the confidence tuple as a weighted sum of the prior and the reported values. The weight factor is the normalized trust index of the sender. The greater the trust associated with the sender, the more inclined the updated confidence in the direction of the reported value. On the other hand, a low trust value of the sender results in the updated confidence value remaining closer to the prior value. Following is the equation used for computing the location confidence, as shown in Algorithm 3.

$$\lambda\mathcal{V}' = SF\left(\frac{T_s}{Tmax}\lambda\mathcal{S} + \frac{Tmax}{T_s}\lambda\mathcal{V}\right) \tag{1}$$

Note that the updated location confidence is a function of both the current confidence tuple of the receiver and the trust assigned to the sender.

c) Map Update: This function updates the terrain type $l[\text{TYPE}]$ to node l within the map $\mathcal{V}[\text{MAP}]$ based on the updated location confidence tuple $\lambda\mathcal{V}'$. It uses the ARGMAX function that selects the type associated with the maximum value of the updated location confidence tuple $\lambda\mathcal{V}'$.

Algorithm 3 Update Functions

```
procedure UpdateSenderTrust($T_S$, $\lambda\mathcal{V}$, $\lambda\mathcal{S}$)
    dissimilarity_index ← $||T_S\lambda\mathcal{S} - T_{max}\lambda\mathcal{V}||$
    similarity_index ← $T_{max}$ − dissimilarity_index
    $conflict \leftarrow CompareOrder(\lambda\mathcal{S}, \lambda\mathcal{V})$
    if conflict TRUE then (Decrement Sender Trust)
        $\Delta T$ ← dissimilarity_index × $(T_{max}/T_S)$
        $T_S \leftarrow T_S - \Delta T$
    else (Increment Sender Trust)
        $\Delta T$ ← similarity_index × $(T_S/T_{max})$
        $T_S \leftarrow T_S + \Delta T$
    return $T_S$
procedure UpdateLocConfidence($T_S$, $T_{\max}$, $\lambda\mathcal{V}$, $\lambda\mathcal{S}$)
    $SF$ ← CalculateScalingFactor($T_s$, $T_{\max}$)
    $\lambda\mathcal{V} \leftarrow SF((T_S/T_{max})\lambda\mathcal{V} + (T_{max}/T_S)\lambda\mathcal{V})$
    return $\lambda\mathcal{V}$
procedure UpdateMap($l$, $\mathcal{M}$, $l$[type], $\lambda\mathcal{V}$, $T_s$, $T_{\max}$, $\lambda\mathcal{S}$)
    $\lambda\mathcal{V}' \leftarrow$ UpdateLocConfidence$(T_S, T_{\max}, \lambda\mathcal{S}, \lambda\mathcal{V})$
    for $l$ in $\mathcal{M}$ do
        $l.l$[TYPE] ← $l$[type]. argmax($\lambda\mathcal{V}'$)
```

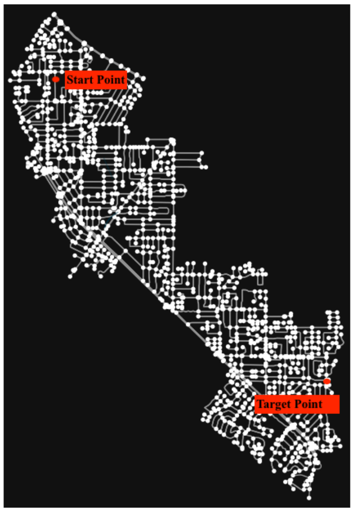

Fig. 2. Simulation Map

VI. Empirical Evaluation

We performed extensive empirical evaluation to justify the viability of DRiFt. To our knowledge, there is no comparable platform to compare with DRiFt directly. To provide a fair assessment, we compared it with DCFM (without resiliency) in malicious and benign settings. Furthermore, note that the cybersecurity challenges only arise because coordination among vehicles in the fleet creates an attack surface for the adversary to disrupt or tamper the communication; consequently, a (drastic) approach to addressing the cybersecurity challenges is for the vehicles to completely eschew coordination and depend only on their local map for navigation. Consequently, to demonstrate the value of coordination, we also perform a suite of experiments to compare coordinated fleet management with no coordination in the fleet. Our performance analysis considers three metrics: the (1) average route length (measured in nodes), (2) the number of vehicles reaching mines, and (3) the average number of obstacles encountered per vehicle (which contributes to delay in completing the mission).

Experimental Setup

Our research involved meticulously designing a simulation platform tailored for DFCM and DRiFt based on realistic terrain models. The main focus was to replicate real-world scenarios for vehicular movements. In terms of technical architecture, our simulation platform was built from the ground up using Python programming language. To add layers of complexity and realism to our network structures, we integrated the *Network X* [21], a Python-based tool for creating, handling, and delving deep into the intricacies of multifaceted network structures.

Given the significance of real-world mapping data, we used the *OSMnx* package [22], that interfaces with the OpenStreetMap API. OpenStreetMap, a continually updated repository of detailed geographical information, is usually used by experts in fields like route planning, traffic analysis, and various transportation endeavors. The combination *OSMnx* and *Network X* allows us to visualize network centrality effectively and save the graph structures that were imported into our simulation platform.

The map used in our simulation platform represented a section of Los Altos, CA. The map is shown in fig. 2 This map encompassed a total of 1351 nodes with 20% of the nodes obstacles or mines, ensuring a rigorous testing environment for our algorithms. Based on Fig 3, 20% total percentage of obstacles and mines provides a sizable challenge without overwhelming the scenario, as the numbers of both mines and obstacles encountered are relatively balanced and manageable. At 35%, DCFM fails as vehicles find it difficult to find alternative paths. In our simulation, we opted for a total of 30 vehicles since adding more vehicles would result in congestion due to the map's constrained space. This, in turn, would reduce the number of safe paths available, as all the vehicles would be moving simultaneously.

Our experiments showed that at 40 vehicles, the map reaches its capacity, and no viable paths remain, demonstrat-

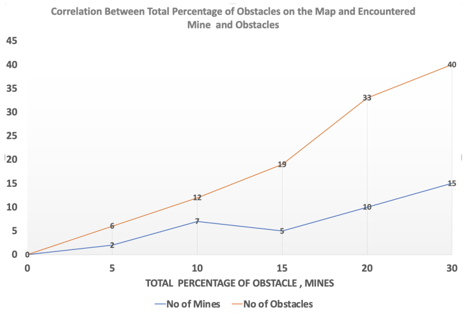

Fig. 3. Correlation Between Total Percentage of Obstacles on the Map and Encountered Mine and Obstacles

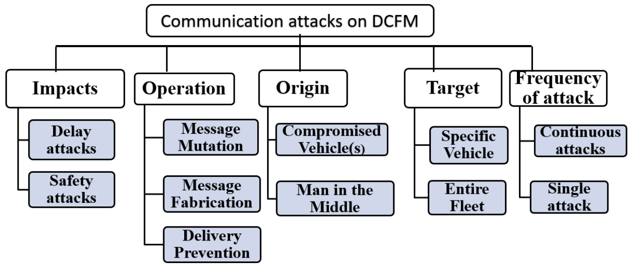

Fig. 4. Attack Taxonomy

ing the map's maximum vehicle capacity. Our simulation also considered a broadcast range, $\mathcal{M}_B$ with a radius of 4 units. The communication mechanism was also equipped with a single-hop re-transmission protocol, ensuring messages were propagated every third time-step, mimicking real-world communication delays and challenges.

A. Attack Taxonomy

Vulnerabilities in vehicular communications manifest themselves in various ways, and the methods used to corrupt them are continually evolving, with new attacks being discovered frequently. This poses a challenge when attempting to cover the attack space comprehensively. However, it is necessary to evaluate safety-critical military applications for robustness against known (N-day) attacks and attacks that are unknown at the time of design/deployment (zero-day attacks).

We address this problem by designing our evaluation strategy to focus on *attack symptoms* rather than *attack mechanisms*. More precisely, we define a taxonomy based on five *classifying features* that are independently sufficient and comprehensive in defining and accounting for *any* communication attacks on DCFM. The idea is inspired by threat modeling approaches in hardware and system security [23], but adapted for the application. In particular, since the adversary is constrained to be capable of disrupting the communications among vehicles, the only choices of the adversary are to (1) mutate an existing message, (2) fabricate a new message, and (3) prevent the delivery of a message. Fig. 4 shows our attack taxonomy. We categorize each attack based on its origin, mode of operation, frequency, target vehicles, and impact on the fleet's vehicles (outcome). We also list various representative attacks as a combination of the five classifying features and evaluate the efficacy of our resiliency approach.

Attack Orchestration

We created specialized attacks based on these features. The attack could originate from a rogue vehicle(s), stationary MITM(s), or Dynamic MITM(s) that moves from one location to another. The Density of these attacks was also variable. The attacks could also emanate from a single source and multiple sources (up to 20% of the total number of vehicles in the fleet) for attacks originating the rogue vehicles.

In practical situations, the likelihood of a significant portion of a fleet being compromised is relatively low. Moreover, a substantial increase in the number of rogue vehicles could result in the failure of DRIFT. This is because DRIFT's effectiveness hinges on reliable communication among trusted vehicles; an excessive presence of rogue vehicles disrupts this essential communication network, undermining the system's functionality.

These vehicle(s) were randomly chosen from the fleet. The attacks could also originate from a single MITM or multiple MITM(s). Every attack is also implemented with predetermined frequencies, which could be discrete, cluster, or continuous. The discrete attacks occur for a single time step, while the cluster attacks last for time steps. The various time steps are chosen randomly. The continuous attacks last for the entire duration of the maneuvers. Message fabrication attacks were implemented by introducing fake and menacing messages into the communication scheme. Mutation attacks involve altering the message payloads in transit. During the attack operation, We classified the normal nodes as obstacles and the obstacles as normal nodes. We also implemented an attack that prevented messages from reaching the intended target. Any vehicle within the broadcast Range will be affected by the various attacks implemented by the framework.

B. Advantages of Coordination

Fig. 5 shows the results on the advantages of coordination. Reducing the number of vehicles reaching mines is the primary desired effect of our DCFM approach. Without coordination, 24 of 30 vehicles in the fleet maneuvers encountered mines.

Coordination also reduced the average number of obstacles faced by the vehicles in the fleet. Note that there is a slight increase in the average route length and the total number of obstacles encountered during coordination. This is because a high number of vehicles encountered mines without coordination and did not complete the mission in the first place. We can infer that coordination is essential for fleet management.

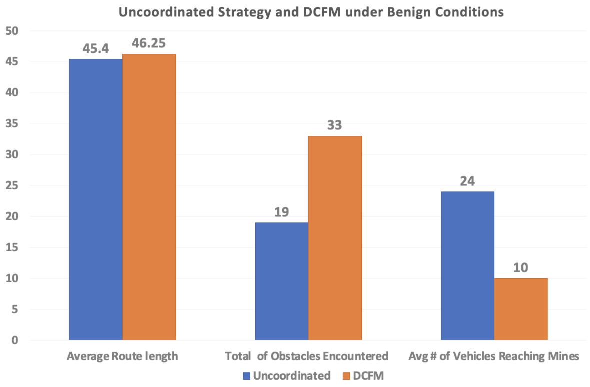

Fig. 5. Effects of DCFM

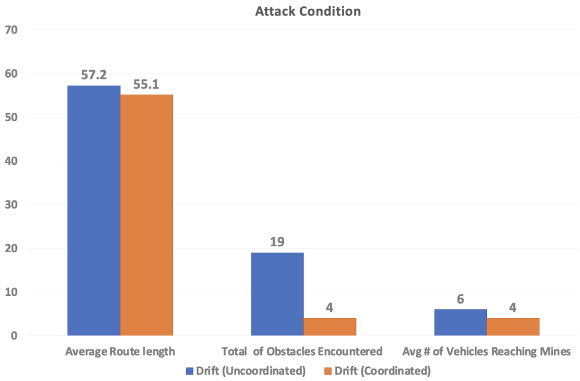

Fig. 6. Resiliency Analysis under Coordinated and Uncoordinated Conditions

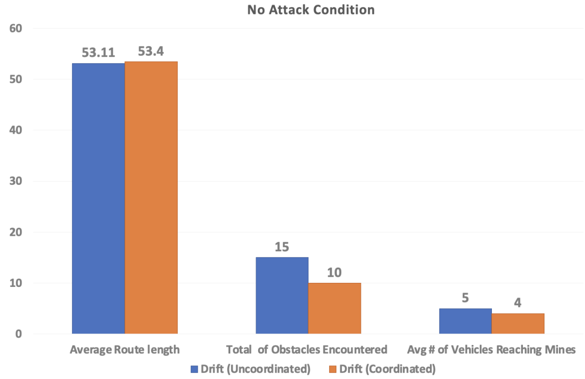

Fig. 7. Resiliency Analysis under Coordinated and Uncoordinated Conditions (Benign conditions)

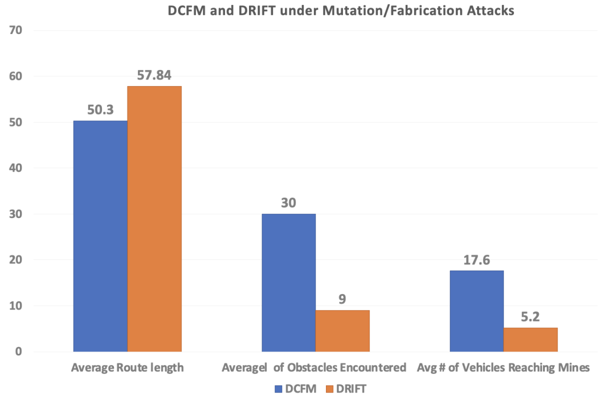

Fig. 8. Comparing DCFM and DRIFT under Mutation/Fabrication Attacks

C. Resiliency Evaluation under Coordinated and Uncoordinated Fleet Management Conditions

We performed experiments to evaluate the resiliency framework under DRIFT and uncoordinated fleet management conditions. Under benign conditions, DRIFT slightly performed better than the uncoordinated fleet management, resulting in a 33% reduction in encounters with obstacles. 5 vehicles were destroyed by mines in the uncoordinated scenario while 4 vehicles reached mine in the coordinated scenario. We also observed that under malicious scenarios, DRIFT performed better with a coordinated fleet when compared with an uncoordinated fleet, as shown in figure 6. Note that the coordinated fleet experienced a 78% reduction in the number of obstacle encounters compared to the uncoordinated fleet. 6 vehicles were destroyed by mines in the uncoordinated fleet, while 4 vehicles was destroyed by a mine in the coordinated fleet. We can thereby infer that coordination between vehicles is necessary to achieve optimal resiliency against attack.

D. Effects of attacks

1) Mutation/Fabrication attacks: On average, DRIFT reduced the number of vehicles that reached mines by an average of 70.4% when compared to the naive application of the DCFM application under all attacks, as shown in figure 8. DRIFT reduced obstacle encounters by an average of 70%. The only drawback of drift is that the average route length increased with the implementation of the DRIFT by an average of 15.0% for the various attacks, as shown in Fig. 8.

2) Delivery Prevention Attacks: On average, DRIFT reduced the number of vehicles that reached mines by an average of 53% when compared to the naive application of the DCFM application under all attacks, as shown in Fig. 9. DRIFT reduced obstacle encounters by an average of 58.8%. The only drawback of DRIFT is that the average route length increased with the implementation of the DRIFT by an average of 7% for the various attacks, as shown in Fig. 9.

VII. Conclusion and Future Work

Coordinated fleet management is a critical and complex connected vehicle application. One key challenge in fleet management is cybersecurity vulnerabilities arising from vehicular communications. In this paper, we presented DRIFT, a resilient, distributed, and connected communication application enabling a fleet of vehicles to perform coordination in the presence of adversaries, subverting the integrity of their communication. DRIFT works by iteratively refining the confidence of a vehicle $\mathcal{V}$ in its estimate of the terrain type of a specific location l in as well as the trust bestowed

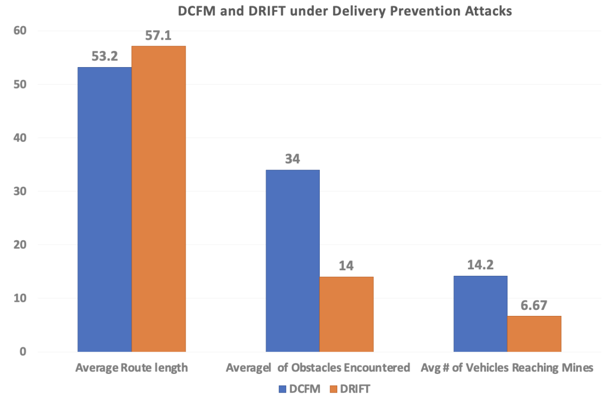

Fig. 9. Comparing DCFM and DRIFT under Delivery Prevention Attacks

on the sender vehicle S providing information regarding l. We provide a thorough empirical evaluation of the approach, demonstrating that DRIFT can successfully achieve resilient coordination to a spectrum of adversaries.

In future work, we will consider extending fleet management to other adversaries, including adversaries that spoof location of vehicles rather than the message payload. We will also explore a more comprehensive evaluation impact of the approach under variations of different design and environmental parameters, such as broadcast range, broadcast intervals, number of vehicles in the fleet, etc.

Acknowledgements: This research has been supported in part by the National Science Foundation under Grant No. CNS-1908549.

REFERENCES

[1] T. Samad, J. S. Bay, and D. Godbole, "Network-centric systems for military operations in urban terrain: The role of uavs," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 92–107, 2007.

[2] S. T. S. Thong, C. T. Han, and T. A. Rahman, "Intelligent fleet management system with concurrent gps gsm real-time positioning technology," pp. 1–6, 2007.

[3] P. Gonzalez-De-Santos, A. Ribeiro, C. Fernandez-Quintanilla, F. Lopez-Granados, M. Brandstoetter, S. Tomic, S. Pedrazzi, A. Peruzzi, G. Pajares, G. Kaplanis *et al.*, "Fleets of robots for environmentally-safe pest control in agriculture," *Precision Agriculture*, vol. 18, no. 4, pp. 574–614, 2016.

[4] J. E. Siegel, D. C. Erb, and S. E. Sarma, "A survey of the connected vehicle landscape—architectures, enabling technologies, applications, and development areas," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2391–2406, 2018.

[5] D. K. Tosh, S. Shetty, P. Foytik, L. Njilla, and C. A. Kamhoua, "Blockchain-empowered secure internet-of-battlefield things (iobt) architecture," *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, 2018.

[6] M. . R. Asaar, M. Salmasizadeh, W. Susilo, and A. Majidi, "A secure and efficient authentication technique for vehicular ad-hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 6, pp. 5409–5423, 2018.

[7] M. S. Sheikh, J. Liang, and W. Wang, "A survey of security services, attacks, and applications for vehicular ad hoc networks (vanets)," *Sensors*, vol. 19, no. 16, p. 3589, 2019.

[8] S. A. Jan, N. U. Amin, M. Othman, M. Ali, A. I. Umar, and A. Basir, "A survey on privacy-preserving authentication schemes in vanets: Attacks, challenges and open issues," *IEEE Access*, vol. 9, pp. 153 701–153 726, 2021.

[9] I. Ali, A. Hassan, and F. Li, "Authentication and privacy schemes for vehicular ad hoc networks (vanets): A survey," *Vehicular Communications*, vol. 16, pp. 45–61, 2019.

[10] D. Gantsou and P. Sondi, "Toward a honeypot solution for proactive security in vehicular ad hoc networks," *Lecture Notes in Electrical Engineering Future Information Technology*, pp. 145–150, 2014.

[11] S. Sharma and A. Kaul, "A survey on intrusion detection systems and honeypot based proactive security mechanisms in vanets and vanet cloud," *Vehicular Communications*, vol. 12, pp. 138–164, 2018.

[12] J. Hortelano, J. C. Ruiz, and P. Manzoni, "Evaluating the usefulness of watchdogs for intrusion detection in vanets," pp. 1–5, 2010.

[13] T. Z hang and Q. Zhu, "Distributed privacy-preserving collaborative intrusion detection systems for vanets," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 1, pp. 148–161, 2018.

[14] M. Erritali and B. El Ouahidi, "A review and classification of various vanet intrusion detection systems," *2013 National Security Days (JNS3)*, pp. 1–6, 2013.

[15] M. Pirani, E. Nekouei, H. Sandberg, and K. H. Johansson, "A game-theoretic framework for security-aware sensor placement problem in networked control systems," *2019 American Control Conference (ACC)*, 2019.

[16] P. N. Brown, H. P. Borowski, and J. R. Marden, "Security against impersonation attacks in distributed systems," *IEEE Transactions on Control of Network Systems*, vol. 6, no. 1, pp. 440–450, 2019.

[17] S. Amin, G. A. Schwartz, and S. S. Sastry, "Security of interdependent and identical networked control systems," *Automatica*, vol. 49, no. 1, pp. 186–192, 2013.

[18] Z. Sun, Y. Liu, and J. Wang, "Game theoretic approaches in vehicular networks: A survey," vol. 13, no. 9, pp. 21–22, Sep 2014.

[19] J. Zhang, C. Chen, and R. Cohen, "Trust modeling for message relay control and local action decision making in vanets," *Security and Communication Networks*, vol. 6, no. 1, pp. 1–14, 2012.

[20] C. A. Kerrache, N. Lagraa, C. T. Calafate, and A. Lakas, "Trouve: A trusted routing protocol for urban vehicular environments," *2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2015.

[21] A. Hagberg, P. Swart, and D. S. Chult, "Exploring network structure, dynamics, and function using networkx," Los Alamos National Lab.(LANL), Los Alamos, NM, United States, Tech. Rep., 2008.

[22] G. Boeing, "Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks," *Computers, Environment and Urban Systems*, vol. 65, pp. 126–139, 2017.

[23] S. Ray, E. Peeters, M. M. Tehranipoor, and S. Bhunia, "System-on-chip platform security assurance: Architecture and validation," *Proceedings of the IEEE*, vol. 106, no. 1, pp. 21–37, 2017.