Location Spoofing Attacks on Autonomous Fleets

Jinghan Yang
Washington University in St. Louis
jinghan.yang@wustl.edu

Andrew Estornell
Washington University in St. Louis
aestornell@wustl.edu

Yevgeniy Vorobeychik
Washington University in St. Louis
yvorobeychik@wustl.edu

Abstract*—A common vision for large-scale autonomous vehicle deployment is in a ride-hailing context. While this promises tremendous societal benefits, large-scale deployment can also exacerbate the impact of potential vulnerabilities of autonomous vehicle technologies. One particularly concerning vulnerability demonstrated in recent security research involves GPS spoofing, whereby a malicious party can introduce significant error into the perceived location of the vehicle. However, such attack focus on a single target vehicle. Our goal is to understand the *systemic* impact of a limited number of carefully placed spoofing devices on the quality of the ride hailing service that employs a large number of autonomous vehicles. We consider two variants of this problem: 1) a static variant, in which the spoofing device locations and their configuration are fixed, and 2) a dynamic variant, where both the spoofing devices and their configuration can change over time. In addition, we consider two possible attack objectives: 1) to maximize overall travel delay, and 2) to minimize the number of successfully completed requests (dropping off passengers at the wrong destinations). First, we show that the problem is NP-hard even in the static case. Next, we present an integer linear programming approach for solving the static variant of the problem, as well as a novel deep reinforcement learning approach for the dynamic variant. Our experiments on a real traffic network demonstrate that the proposed attacks on autonomous fleets are highly successful, and even a few spoofing devices can significantly degrade the efficacy of an autonomous ride-hailing fleet.

I. Introduction

Autonomous driving has the potential to transform mobility. A common (although by no means universal) vision for this transformation is that autonomous vehicles would come to serve in large fleets as part of *ride-hailing* services, ultimately obviating the need for people to own and drive cars (3; 14). Indeed, Waymo has already begun a limited deployment of a fully autonomous ride hailing service (13). However, given the potentially transformative impact such services may have on communities, it is imperative that we comprehensively understand their potential limitations.

One class of such limitations come in the form of *security vulnerabilities*, whereby a malicious party attempts to subvert the ride hailing service, either to specific ends, or simply to wreak havoc. We consider one important class of such vulnerabilities in which attackers can tamper with the vehicles' perception of their location; we refer to these as *location spoofing attacks*. One means of actualizing such attacks is to use *GPS spoofing* which interferes with the vehicles GPS signal, causing the vehicle to have an incorrect belief about its current position (2; 15; 16; 19; 21; 22). Attacks of this kind involve a spoofing device placed near, or in, a target vehicle. Most prior works focus on the technical aspects of implementing the attack itself. We take such *GPS spoofing* attacks as a given, and ask a broader systemic question: is it possible to leverage GPS spoofing attacks to significantly impact a ride-hailing service at scale with a limited number of spoofing devices?

To answer this question, we consider three threat models that involve placing a limited number of location (GPS) spoofing devices in a traffic network. In the first, the spoofing devices are placed in a set of fixed locations (intersections) in the traffic network, and each device spoofs a fixed target location; we refer to this as the *static-static* model (static device locations with static spoofed locations). In our second model, the spoofing device locations are still fixed, but the target locations being spoofed can now be time-varying (the *static-dynamic* model). Finally, our third model allows both the devices and the locations to vary with time, for example, with devices being transported in malicious drones. We consider two possible attack goals: 1) maximizing overall delay of servicing requests, and 2) maximizing the number of failed (not completed) requests, where a failed request is one in which a passenger is dropped off at an incorrect location.

We begin with a negative result from an attacker's perspective (positive from the vantage point of a fleet manager): even in the static-static setting, computing an optimal location spoofing attack is NP-hard, even when one targets a single vehicle and has an unlimited spoofing budget. Next, we present algorithmic approaches for each of the three threat models. For the static-static model, we propose an integer linear programming (ILP) formulation for both service delay and service failure attacks. In addition, we present a simple greedy heuristic approach for the service delay attack in this setting. We then adapt the ILP formulation from the static-static to the static-dynamic setting, blending it with a greedy heuristic approach for selecting spoofed locations. Finally, we present a deep Q-network (DQN)-based reinforcement learning approach for learning a spoofing attack strategy in the dynamic-dynamic setting.

We evaluate the proposed attack methods through extensive experiments. Our results demonstrate that attacks can indeed be highly effective, despite the negative worst-case hardness

Symposium on Vehicles Security and Privacy (VehicleSec) 2023
27 February 2023, San Diego, CA, USA
ISBN 1-891562-88-6
https://dx.doi.org/10.14722/vehiclesec.2023.23031
www.ndss-symposium.org

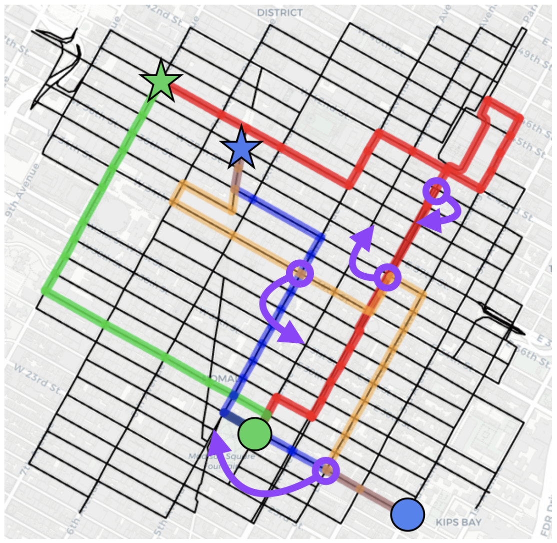
Fig. 1: Routes from starting location (circle) to destination (star) of two fleet vehicles with no spoofing (green and blue) and with 4 spoofing devices (red and orange) in *OpenStreetView*. The traffic graph (radius 1000) is marked in black; spoofing locations and effects are marked as purple arrows.

results. We show that our proposed approaches outperform several baselines, often by a significant margin, particularly as the problem size increases.

In summary, we make the following contributions:

- We systematically study the effects of GPS spoofing attacks on multi-vehicle self-driving fleets, under two types of attacking objectives (maximizing delay and maximizing number of incomplete trips) and three paradigms of spoofing capabilities (static-static, static-dynamic, dynamic-dynamic).
- We show that the attacker's problem is intractable for each combination of spoofing capabilities and objective.
- Despite this intractability we provide efficient solutions for developing attacks in practice: an ILP formulation for the static-static case, an effective greedy heuristic for the static-dynamic case, and a deep reinforcement learning approach in the dynamic-dynamic case.
- We experimentally demonstrate the effectiveness of each attack method on a simulated traffic network of New York City (sourced from *OpenStreetMap* and *Uber Movment*), and demonstrate that even a few spoofing devices can significantly impact the performance and reliability of a large autonomous vehicle fleet.

II. Related Work

In recent years extensive research, both academically and commercially, has focused on developing fully autonomous vehicles (4; 6; 1). Many of these autonomous driving paradigms can be thought of as consisting of two systems: a low-level system which is responsible for controlling the vehicle and effectively navigating the roadway, and a high-level system which is responsible for routing the vehicle to its destination. Our focus will be primarily on the high-level system. Within this high-level navigation system GPS is the de facto means of determining vehicle location. When navigating via GPS location information, the vehicle possesses a receiver which accepts GPS signals which indicate the vehicle's current location. Recent work has demonstrated that the GPS signal received by the vehicle can be maliciously altered by spoof devices (22; 2; 7; 2; 15; 16; 19; 21; 22). In GPS spoofing, the attacker transmits fabricated GPS signals with greater signal power than the authentic ones, thus causing the victim receiver to lock onto the attacker's signals, rather than the authentic signal, and resolve positions controlled by the attacker. GPS spoofing has been proven feasible theoretically (8) and empirically (22; 16). More importantly, it has been demonstrated that autonomous driving vehicles are vulnerable to GPS spoofing attacks, such as Tesla cars (10), in the physical world. Further, (22; 9) has shown that modern spoofers are cheap to manufacture.

Another orthognal line of research has investigated the capability of using GPS spoofers to reroute autonomous vehicles. In particular, (22) has developed a spoofing attack which is capable of successfully rerouting a victim car such that it arrives at an incorrect destination. This work conducted physical experiments in New York City in which they successfully rerouted a vehicle and reached a location that is different from its destination, thus demonstrating that GPS spoofing attacks are physically realizable. In their design, the attackers knows the vehicle's destination, the vehicles location at each time timestep, and the vehicle's routing algorithm. Spoofers are placed at every intersection and the attacker designs a signal for each spoofer such that when the vehicle's routing system uses the designed signal, the vehicle is rerouted to the attacker's target location. These and similar GPS spoofing attacks typically impose two constraints: first, they require a spoofing device to be close to the target vehicle, and second, they require the spoofed location to be reasonably close to the vehicle's true location. The latter constraint is intended to create subtle attacks, which are more difficult to detect.

While GPS spoofing attacks have been demonstrated to consequentially impact autonomous driving, prior work has only considered single-vehicle attacks. In contrast, we consider attacks on a multi-vehicle ride-hailing fleet involving multiple spoofing devices. Our attacks share some structure with variants of multi-agent routing problems (12; 17; 18), although many technical details differ from our problem structure differ.

III. Model

We begin with a model of a ride-hailing system, in which K vehicles traverse a traffic network $G = (V, E)$, where V is the set of nodes representing intersections and E is the set of edges connecting the intersections. We use the set $\mathcal{N}(i)$ to denote the neighbors of node $i \in V$. Each edge $e_{ij} \in E$ in the traffic network G is associated with a travel time $c_{i,j}$, which we normalize without loss of generality to be in $[0, 1]$. We identify each vehicle with an index $k \in \{1, \dots, K\}$. Vehicle k is associated with a current location s_k and a destination d_k. Further, we assume that each vehicle k follows the shortest path from any location $v \in V$ to its destination d_k. We assume that both G and the edge costs are fixed (e.g. the traffic of each street is roughly invariant over time), and we can therefore pre-compute a shortest path from each starting point and destination.

Our model of spoofing effect on the vehicle routes is based on the model proposed by Zeng et al. (22), who constructed a physically realizable spoofing attack which effectively manipulated a single vehicle. In this model, the attacker chooses the intended effect of a spoofing device on a target vehicle with the objective of inducing a desired behavior, e.g. maximally deviating from the target destination. In our setting, we focus on a fleet of autonomous vesicles, each of which traverse the network that may be affected by the same spoofing device. Consequently, the same spoofing device and effect will be reflected in any vehicle that comes within the proximity of the device. We capture the behavior of each such vehicle by a matrix M, where $M_{i,j} = \alpha$ if a vehicle at node $i \in V$, heading to destination j, will take action α, which can be a high-level action such as which next road segment to take, or a low-level action, such as a steering angle. In our experiments, we focus on the latter. Note that the matrix M is independent of the identity of the vehicles (indeed, it can also represent any deterministic behavior of vehicles, not merely shortest paths.[1])

As in Zeng et al., we assume that the attacker knows the network G, edge costs $c_{i,j}$ and vehicle behavior matrix M. These are mild assumptions: G is typically public knowledge, $c_{i,j}$ can be obtained in nearly real-time from mapping applications, and M is vehicle agnostic and can be pre-computed by the attacker. In addition, the attacker knows the current state of the K fleet vehicles, for example, as a consequence of a separate cyber-attack on the fleet management service.

The attacker is endowed with a collection of B spoofing devices. Each device can be placed at a location $i \in V$, and spoof an alternative *target* location $j \in V$ chosen by the adversary. As a consequence of such an attack, a vehicle at location i erroneously perceives that it is actually at location j. It will be useful to represent the attacker's decision by a binary variable x, where $x_{i,j} = 1$ iff there is a location spoofing device placed at location i which targets location j.

In general, we allow the attack to vary with time, in which case x_t will denote the attacker choice at time t. In particular, we consider three attack settings that represent three distinct attack capabilities, in increasing order of strength:

1) ***static-static***: both the spoofing device and target locations are fixed (independent of time),
2) ***static-dynamic***: spoofing device locations are fixed, but target locations can vary with time,
3) ***dynamic-dynamic***: both the spoofing device locations and target locations can vary with time.

The former two threat models simply involve placing the spoofing devices at target intersections. The third requires an additional capability that these devices are placed on adversarial mobile vehicles, such as cars or drones. In order to capture any difference in the time scale of vehicle and spoofing device motion, we assume that the devices can change location every t steps for $t \geq 1$ (i.e., they are no faster, but could be somewhat slower, than the vehicles).

We consider two attack objectives:

1) **service delay attack:** maximize the total travel time of the vehicles in the fleet, subject to the constraint that requests are ultimately correctly completed (that is, the rider is not dropped off at the wrong destination), and
2) **service failure attack:** maximize the number of requests in which the rider is dropped off at the wrong destination.

In the case of the first attack, the constraint that each request is ultimately completed serves as a form of stealth, as we also suppose in this context that spoofing devices are on for a finite time duration, and turned off thereafter. The second attack, in contrast, entails vehicles actually believing that they had reached their target destinations, but in fact they have reached locations in which a spoofing device is currently active and spoofing to their dropoff location.

We begin by first noting that the spoofing problem is computationally intractable in general.

Theorem 1. *Maximum delay is NP-hard even for a single fleet car, a single request, unlimited spoofers and uniform congestion for both the online and offline settings, as well as for each of the three spoofing paradigms.*

Proof. We can reduce from Hamiltonian cycle. Let G be an unweighted connected graph. An instance of our problem can be constructed by assigning every edge in the graph weight 1 and creating two nodes called s and d where s has an outgoing edge with weight 1 to each node in G and each node in G has an outgoing edge with weight 1 to d. Let s and d be the fleet car's starting location and destination respectively. A second set of nodes and edges is created such that for each edge (u, v) in G a new node u' is created and the edge (u', d) is created to correspond to (u, v) so that if the vehicle is at node u, but has perceived location u' it will move to v since it thinks it is moving to d. Since congestion is uniform, the spoofing budget is unlimited, and each edge (u, v) has a corresponding edge (u', d) the attacker has arbitrary control over the vehicle's movement and thus the attacker is selecting the vehicle's path with the constraint that the path must start at s and end at d.

Suppose that a solution to Hamiltonian cycle on G is given, then starting from some node u and ending at some node v we have a path which reaches every node in G exactly once. If the adversary where to choose to send the vehicle from s to u and then along the Hamiltonian cycle solution path to v and then from v to d the vehicle's travel time would be $n + 1$. Since each node can be visited at most once by the car, otherwise the car will never reach its destination, and traveling between any two neighbor nodes has cost 1, the maximum travel time the car could have is $n + 1$. Therefore given a solution to Hamiltonian cycle, the spoof attack with the largest travel time can be created. If instead we had an optimal spoofing attack,

[1]Our results and attack framework can be applied to any routing deterministic routing system which is 1) agnostic to vehicle id and 2) assumes that congestion is static.

then since traveling between any two nodes has cost 1 and the agent cannot repeat nodes, otherwise they will not reach d, the number of nodes visited is equal to $t+1$ where t is travel time. Since the agent can only reach nodes s, d and all nodes in G, and must reach nodes s and s, if travel time is t then the agent reached $t-1$ nodes in G. Thus if the spoof attack returns travel time $n+1$ then the vehicle reaches n nodes in G exactly once, and G has a Hamiltonian cycle. Therefore, given a solution to GPS spoofing a solution to Hamiltonian cycle can be found, and given a solution to Hamiltonian cycle a solution to GPS spoofing can be found. □

Despite the hardness result, we next proceed to develop effective algorithms for both static and dynamic spoofing.

IV. Design of GPS Spoofing Attacks on Autonomous Fleets

A. Static Location of Spoofing Devices and Spoofing Targets

We begin by considering static-static attacks in which B spoofing devices are placed at fixed locations in the traffic network, and each is configured to spoof a fixed target location. We first develop an integer linear programming approach when the attack goal is to maximize service delay (with the constraint that the destination is ultimately reached). Subsequently, we show that the second problem of maximizing service failure is tractable, and present an efficient algorithm for it.

1) Service Delay Attack: Recall that when a spoofer and a fleet vehicle are on the same node in the graph, the fleet vehicle will perceive its location to be a different node in the graph specified by the spoofer. Our goal is to find the placement and effect of B spoofers, resulting in the maximum delay to the K fleet vehicles. We adopt two constraints on GPS spoofing attacks proposed by Zeng et al. (22) which ensure physical realizability. The first constraint is on the maximum spoofing distance from the device location. The second constraint prevents the action α induced by the spoofing device from being infeasible in the *actual* location that the vehicle is in, such as turning left when no left turn is available.

Recall that x is a matrix representing spoofing device locations (rows) and targets (columns); since both are static, x does not depend on time. Let $\tau(i,j,x)$ denote the travel time from location i to destination j if spoofing devices are placed and configured according to x. Then the attacker's goal is to solve the following optimization problem: $\max_x \sum_{k=1}^{K} \tau(s_k, d_k, x)$, that is, to choose spoofing device placement and configuration x that maximizes total travel time for all vehicles from their starting locations to their respective destinations. Note that travel time is also implicitly a function of the movement tensor M which represents actions vehicles take in each location and for each possible destination. Critically, when we spoof locations, the net effect is that the vehicles are choosing their actions using an induced movement tensor M', where the entries in M corresponding to a location i are replaced with entries corresponding to a spoofed location j (when a spoofing device is placed at i and targets j).

We propose an ILP formulation for this problem. In this formulation, we construct the movement matrix induced by the spoofing devices M', as well as the associated shortest paths encoded by y for each vehicle, as an explicit function of the binary spoofing decision matrix x, where $x_{i,j}=1$ if we place a spoofing device at location i and spoof location j, and $x_{i,j}=0$ otherwise. As mentioned above, we constrain spoofing distance from the location of the spoofing device, and impose a constraint that a vehicle action induced by the GPS spoofing device is feasible in its actual location. We encode these constraints using a 3 dimensional tensor $F^{\{V,V,A\}}$, and construct this *feasibility* tensor F beforehand, where A is the set of possible vehicle actions. Specifically, $F_{i,j,\alpha}=1$ when both (1) the distance between location i and location j is within the maximum spoofing radius of a spoofer, and (2) it is feasible to take action α at location i. However, if either constraint doesn't hold, $F_{i,j,\alpha}$ is set to be -1. Let $y^k_{i,j}$ be binary variables that represent whether the edge (i,j) is traversed by the vehicle k. Armed with this notation, formulation (1) gives an integer program that computes the optimal static-static attack.

$$\max_{x,y,M'} \sum_{k\in K}\sum_{i\in V}\sum_{j\in\mathcal{N}(i)} y^k_{i,j}\cdot c_{i,j} \tag{1a}$$

$$\sum_{j\in\mathcal{N}(i)} y^k_{j,i} = \sum_{u\in\mathcal{N}(j)} y^k_{j,u} \qquad \forall k\in\{1,\dots,K\}, i\in V: i\neq s_k, d_k \tag{1b}$$

$$\sum_{i\in\mathcal{N}(s_k)} y^k_{s_k,i} = \sum_{i\in\mathcal{N}(d_k)} y^k_{i,d_k} = 1 \qquad \forall k\in\{1,\dots,K\} \tag{1c}$$

$$y_{i,j}+y_{j,i}\leq 1 \qquad \forall\, i,j\in V \tag{1d}$$

$$M'_{i,u,\alpha} = \sum_{j\in V} M_{j,u,\alpha}x_{i,j}F_{i,j} + M_{i,u,\alpha}\cdot\left(1-\sum_j x_{i,j}\right) \forall i\in V, \alpha\in A \tag{1e}$$

$$y^k_{i,i+\alpha} = \sum_{j\in\mathcal{N}(i)} y^k_{j,i}M'_{i,d_k,\alpha} \qquad \forall k\in\{1,\dots,K\}, i\in V\setminus s_k, \alpha\in A \tag{1f}$$

$$y^k_{s_k,s_k+\alpha} = M'_{s_k,d_k,\alpha} \qquad \forall k\in\{1,\dots,K\}, \alpha\in A \tag{1g}$$

$$\sum_{i,j} x_{i,j}\leq B; \quad \sum_j x_{i,j}\leq 1 \quad \forall i\in V. \tag{1h}$$

The objective maximizes the total travel cost (time) incurred by all vehicles. Constraints (1b)-(1d) are standard network flow constraints. Constraint (1e) effectively copies the entries from M to M' corresponding to the spoofed locations (the first term on the right-hand-side), and leaves unspoofed entries as given by M for locations without a spoofing device (the second term on the right-hand-side). Shortest path actions y are then computed using the movement tensor M' induced by the attack in Constraints (1f) and (1g). Finally, Constraints (1h) include the constraint on the maximum number of spoofing devices, and ensure that only one location can be spoofed by any device. While the ILP has constraints which are bilinear, linearization is straightforward (given in the Supplement).

2) Service Failure Attack: To maximize the number of failed requests is to maximize the number of vehicles that mistakenly drop off a passenger at the wrong destination. This clearly entails two features of any spoofing strategy: 1) a device must be placed in a location which some vehicle reaches at some point during its commute, and 2) the device spoofs a destination for these vehicles. This suggests the following simple SF-GREEDY algorithm for choosing spoofing device locations and targets:

1) Partition the vehicles into groups such that each group G_{ld}, $l \neq d$, contains all the vehicles having two properties: a) they share the destination d and b) their paths to destination induced by M intersect at location l.
2) Choose B largest groups G_{ld} with distinct locations l (that is, if there are two groups G_{ld} and $G_{ld'}$ for some location l, we choose the larger group and omit all others). For each chosen group, place the spoofing device in location l with a spoofing target d.

This algorithm clearly runs in polynomial time. The next result, which is proved in the Supplement, shows that this algorithm yields an optimal solution.

Theorem 2. SF-GREEDY *yields an optimal solution to the service failure attack in the static-static setting.*

Proof Sketch. First, note that if we have multiple vehicles traversing l, any spoofing sequence starting with the device placed in l will group them together, since we constrain that only a single location can be spoofed by any spoofing device (and no more than one such device can be placed at an intersection). Thus, we would ultimately only cause service failure for the the largest subgroup of vehicles who traverse l that share a destination. But this means that all of these vehicles can simply be directly pointed to their destination by the spoofing device at l. Consequently, there is always an alternative optimal solution in which each spoofing device at location l simply points to the destination d of the largest group that shares the destination, yielding the structure of the solution in SF-GREEDY. □

B. Static Location of Spoofing Devices with Dynamic Spoofing Targets

In the case of static spoofing device location, but dynamic spoofing (effect can update over time), we can no longer directly apply either the ILP when the objective is to maximize delay, nor the greedy algorithm above if the objective is to maximize service failure. We propose the following heuristic for this case when the objective is to maximize delay. First, we compute optimal spoofer placement via the the ILP (1), assuming that target locations are also static. We then greedily and dynamically modify the spoofing targets in each time step, choosing the targets that maximize the marginal increase to delay. If the objective is to maximize the number of service failures, we replace the first step above with *SF-Greedy*, while the second step greedily maximizes marginal increase to the number of service failures.

C. Dynamic Location of Spoofing Devices and Spoofing Targets

In the fully dynamic case, instead of making only a single decision regarding spoofing location, the attacker now makes a sequence of decisions within a time window. At each decision time, the attacker can change both the spoofing effect and spoofing location by moving a spoofing device in one of a collection of locations L (e.g., up, down, left, right, and no change, if the traffic network is a grid). This can be implemented, for example, if the spoofing devices are located in malicious vehicles or drones. Figure **??** illustrates the process. We will refer to a spoofing device (which is now mobile) as an agent in this section. The underlying problem in this setting is a dynamic (discrete time) decision about the location and spoofing target of *each agent* (each spoofing device) through a given horizon, and extremely complex combinatorial optimization problem. We approach this problem using a multi-agent reinforcement learning (MARL) paradigm. Specifically, each agent independently learns to decide which location to visit and what the spoofing effect (i.e., spoofing target location) is at each decision time. A key challenge in applying MARL in our setting is how to design individual agent rewards, as well as the state space representation, to enable effective learning. Next, we present our solution to both of these issues.

a) Rewards: We designed delay attack reward and service failure reward for the each of the attacking objectives discussed in Section III. In the attack aiming to maximize the overall delay of the fleet, which we denote by η, the objective is

$$\eta = \sum_k \sum_t (\tau_t'^k - \tau_t^k),$$

where $\tau_t'^k$ is the travel time after the attack while τ_t^k is the travel time before the attack. The incremental delay at time t of car k is $\rho_t^k = \eta_t' - \eta_t$. This incremental delay of car k is caused by spoofer j, if the spoofer j is directly next to car k (i.e., they share the intersection). We use the notation $z_t(k, j)$ as a delay indication variable for agent j, $z_t(k, j) = 1$ if agent j shares the intersection with car k (i.e., is able to spoof the location for this car). We use the incremental delay caused by agent j as the reward for this agent at time t, i.e.,

$$\rho_t^j = \sum_k z_t(k, j) \cdot \rho_t^k$$

In the case of service failure attack, the reward is simpler: if the agent succeeds in causing service failure in step t, it receives the reward of 1; otherwise, it receives reward of 0.

b) State Representation: In the dynamic-dynamic case, we construct state representation based on two types of information relating to the traffic network: *invariant*, consisting of information which does not change from one time-step to another (e.g., traffic network structure), and *real-time*, consisting of information which changes over time (e.g., vehicle locations). This representation is deigned to be applicable to any transformer-based architecture.

The *invariant information* is represented in the form of node features, which capture important and static aspects of each node $i \in V$. We refer to the matrix of all node features as

$\boldsymbol{\xi}$, where $\boldsymbol{\xi}_i$ denotes the features of node i. Specifically, each node i is mapped to two types of features. First, we include information about costs (e.g., congestion) of all out-edges incident to i, which in our case of grid networks includes (up to) four incident edges. This implicitly assumes (as does our model above) that spoofing has no effect on edge costs (e.g., when the size of the fleet is small relative to overall traffic). Second features of node i include movement features consisting of $M_{i,j,d}$, flattened as a vector, over all values of destinations j and movement directions d.

Real-time (or time-evolving) features consist of features corresponding the vehicles' behavior, and action of other agents (i.e., other spoofing devices), at each time t. Recall that in the static-static setting, M' denoted the movement tensor of vehicles after spoofing. Here, as this quantity now changes over time, we denote the corresponding tensor by M'_t; this is our first set of real-time features. Next, we associate the following real-time features with each vehicle k:

1) δ^k_t, which is the vehicle's current destination (which we now allow to evolve in time); this allows us to know which portion of the movement tensor M' the vehicle is effectively using to determine its next action,
2) h^k_t, the time that the vehicle requires to complete its current action (i.e., to travel to the next node); again, note that this generalizes our model by allowing this to vary by vehicle and edge, as well as in time,
3) l^k_t, the current location of the vehicle, and
4) M'_t, the current move instruction, and
5) ζ^k_t, the vehicle's path induced by the current positions of spoofing devices and their spoofing targets.

Finally, we include as a feature the location of the spoofing agent j at time t, denoted by l^j_t. We denote the real-time features by $\mathbf{s}_t$.

V. Experiments

A. Experiment Setup

For our experiments, we use a Manhattan, NY traffic network obtained using *OpenStreetMap* (5). Following the convention in (22), we construct a directed graph $G = (V, E)$ to represent the road network. Each node represents an intersection, and has a unique id as well as coordinates in the constructed graph. Adjacent nodes $i, j \in V$, which are intersections connected by a road, are themselves connected by a directed edge $e_{i,j} \in E$. Additionally, each pair of edges e_{ij}, e_{jk} has a turning angle α_{ijk}, which gives the angle required for a car to turn from intersection j (when arriving from road e_{ij}) onto road e_{jk}.

In addition to the purely geographic information about the traffic network obtained from *OpenStreetMap*, we add congestion to each edge $e_{i,j}$ using real traffic congestion data for each road segment obtained from *Uber Movement* (20), an open source platform that provides real time traffic flow data collected by Uber users. Specifically, we use the average congestion for each Friday at 5 pm in March, 2020; this is the most recent traffic data available for New York City. *Uber Movement* and *OpenStreetMap* use the same convention to label both locations and road segments, and the data from these can therefore be directly integrated.

We conduct experiments in a geographic area centered at 350 5th Ave, Manhattan, New York, with a radius of 500 or 1000 meters; Figure 1 gives this network (black lines) for a 1000m radius. Finally, we build a traffic simulator based on the downloaded traffic network and the traffic flow information. In this simulator, requests are assigned to fleet vehicles by the routing center, thereby defining each vehicle's destination as either the pickup or dropoff location of the assigned request. Requests are spawned uniformly at random across the network.

Evaluation Metric: We use *delay ratio* to evaluate the effectiveness of the attack, defined as $\omega = \frac{\tau' - \tau}{\tau}$, where τ is the original travel time and τ' the travel time in the presence of malicious spoofing. We conduct experiments varies from 1 to 20 fleet sizes with a spoofing budget of 1, 5, or 10.

Baselines: We compare our method to two baseline methods. First, *random spoofing*, where the locations and targets of spoofing devices are chosen randomly (in the dynamic settings, these are chosen randomly in each time step). The second baseline is *greedy spoofing*. In the static-static case, greedy spoofing consists of iteratively choosing both the location and target of each spoofing device to maximize the marginal increase in delay. In the static-dynamic case, spoofing devices are placed at locations that maximize the number of intersecting vehicle paths, while the spoofing targets (chosen at every time step) are selecting to maximize the increase in delay. In the dynamic-dynamic case, each spoofing device moves greedily at each iteration to minimize the distance to the closest non-spoofed fleet vehicle, while spoofing effects are chosen to maximize the marginal increase in total delay. When using the random spoofing strategy in the dynamic-dynamic case, the attacker randomly moves the spoofers and randomly selects their spoofing effect.

B. Results

1) Static-Static Case: Recall that in the *static-static* case, both the locations and targets of spoofing devices are fixed. In the modified driving environment, if a GPS spoofing device is placed at location i and spoofs location j, any fleet vehicle passing through i will receive the modified GPS signal, thus perceiving its current location as j. This discrepancy between physical and perceived location may result in a detour from a vehicle's intended path. Such detours may cause the car to arrive at its destination late, or be unable to complete the request.

We begin by showing the results for the maximum-delay attack, in which the attacker aims to maximize the fleet's average delay while ensuring that the requests are still completed. Table I shows the results on different traffic network sizes. In this table we see that even with a single spoofer, delay attacks can successfully increase the fleets average travel time. As to be expected, delay ratio decreases as the number of fleet vehicles relative to the number of spoofers, increases. Moreover, we see that our proposed method outperforms random in all cases, and outperforms greedy when the spoofing budget is greater

Spoofing Radius	Spoofing Budget	#Target Cars	Travel Time	**Proposed** Delay Ratio	**Greedy** Delay Ratio	**Random** Delay Ratio
1	1	1	200	**0.11**	**0.11**	0.0
1	1	5	262	**0.02**	**0.02**	0.0
1	5	5	262	**0.5**	0.3	0.00
1	5	10	242.7	**0.3**	0.3	0.01
2	5	5	262.2	**0.9**	0.56	0.00
2	5	10	242.7	**0.75**	0.34	0.00
1	10	10	242.7	**0.93**	0.5	0.00
1	10	20	252.79	**0.72**	0.37	0.02
2	10	10	242.7	**1.1**	0.7	0.01
2	10	20	252	**0.9**	0.75	0.02

TABLE I: The average travel time in unmodified driving environment and delay ratio induced by spoofers in the *static-static* case in a *dist-1000 meters* traffic network. The travel time unit is second.

Spoofing Radius	Spoofing Budget	#Target Cars	Travel Time	**Proposed** Delay Ratio	**Random** Delay Ratio
1	1	1	200	**0.4**	0.01
1	1	5	262	**0.22**	0.00
2	1	1	200	**0.5**	0.01
2	1	5	262	**0.28**	0.00
1	5	5	262	**0.35**	0.04
1	5	10	242.7	**0.92**	0.06
2	5	5	262	**1.9**	0.05
2	5	10	242.7	**1.2**	0.09
1	10	10	242.7	**0.17**	0.03
1	10	20	252.79	**0.16**	0.02
2	10	10	242.7	**0.17**	0.03
2	10	20	252	**0.16**	0.02

TABLE II: The average travel time in unmodified driving environment and delay ratio induced by spoofers in the *static-dynamic* case in a $dist - 1000$ traffic network.

than 1 (note that for $B = 1$ greedy is optimal). This trend persists for other cases outlined in the Supplement.

2) Static-Dynamic Case: In the static-dynamic case, the attacker can update the spoofing effect over time. We propose a two-step approach to solving this case (The Algorithm is defered to the Appendix). First we run static-static ILP formation 1 to get location of each spoofer. Second, at each timestep we greedily select the spoof effect, for each spoofer, which results in the maximum increase to travel. Table II shows delay ratio for different numbers of victim cars, spoof budget, network size. We defer the results of network size 500 to the appendix. Our method outperforms random selection significantly in every case.

3) Dynamic-Dynamic Case: In the dynamic-dynamic case, the attacker is able to move the spoofing devices and change the spoofing targets at each timestep. A problem instance is specified by 1) the starting location of each agent (spoofing device), 2) the starting location of each fleet vehicle, and 3) the pickup and dropoff locations of the requests. We refer to the combination of these three elements as the configuration of the problem.

TABLE III: The delay ratio in the *dynamic-dynamic* case in a $dist - 500$ traffic network with induced by spoof devices with spoofing raidus 1.

Spoofing Budget	#Target Cars	Travel Time	**Proposed** Delay Ratio	**Greedy** Delay Ratio	**Random** Delay Ratio
1	1	200	**0.90**	0.89	0.03
5	5	262	**2.0**	1.2	0.09
5	10	242.7	**1.11**	0.67	0.05
10	20	252	**1.0**	0.78	0.09

TABLE IV: The delay ratio in the *dynamic-dynamic* case in a $dist - 1000$ traffic network induced by spoof devices with spoofing radius 1.

We train on examples with random problem configurations, and evaluate our proposed method and the baseline methods, on a randomly generated test set of 500 problem configurations. We make use of the reinforcement learning paradigm known as Target Deep Q Networks in which a deep neural network is used to approximate the Q-values of each state action pair. Here a state, the representation (described in Section IV-C0b) consists of the traffic network, requests, fleet locations, and adversarial car locations. Each action consists of both a direction for the adversarial device to move as well as a spoofing target. Stochastic gradient decent is used to update the model. We provide full implementation details and hyperparameter choices in the Supplement.

We see that in all cases, our proposed model outperforms random and greedy.In particular, as the number of agents and fleet vehicles increases, the proposed method outperforms greedy by an increasing margin. This is to be expected as the dynamics of the problem become more subtle as the number of agents and fleet vehicles increases. Moreover, note that spoofing in the dynamic-dynamic case is more effective (causing larger delay) than the static-static case. The ability to dynamically update the spoofer device locations and targets greatly improves the efficacy of the attacks.

Finally, we evaluate our dynamic-dynamic attacks in an online setting. In this setting, new requests are dynamically spawned over time by uniformly selecting a pickup and dropoff location in the traffic network. Fleet assignment to requests is also dynamically updated once a vehicle has completed its current request. In our simulations, we adopt the assignment strategy proposed by (11). On average, each car takes 50 requests. Complete details of the the online setting are provided in the Supplement. As shown in Table IV, the delays induced by our RL-based attacks in the online setting are comparable to those induced when vehicle requests (i.e., destinations) are statically assigned. This corresponds to a 100-180% increase in delay.

VI. Conclusion and Future Work

We systematically investigate the impact of malicious GPS spoofing on ride-sharing. We propose three models of multi-device GPS spoofing in urban autonomous ride-hailing settings: 1) static spoofing device locations and targets, 2) static locations, but dynamic spoofer targets, and 3) allowing both spoofing locations and targets to change over time. We then propose algorithmic approaches for each attack setting. We observe that each of the three classes of attacks can cause significant increases in fleet travel time or significant numbers of failed requests. There are many interesting extensions to this work that merit further exploration. In particular, one can consider the effect of spoofing on network congestion, which will in-turn impact the travel time of the fleet. Moreover, the success of GPS spoofing attacks we propose gives rise to the need for ride hailing and routing frameworks that are robust to spoofing, as well as other information failures.

References

[1] "Waymo has launched its commercial self-driving service in phoenix — and it's called 'waymo one'."

[2] J. Bhatti and T. E. Humphreys, "Hostile control of ships via false gps signals: Demonstration and detection," *NAVIGATION, Journal of the Institute of Navigation*, vol. 64, no. 1, pp. 51–66, 2017.

[3] L. Burns and C. Shulgan, *Autonomy: The Quest to Build the Driverless Car—And How It Will Reshape Our World*. Ecco, 2018.

[4] cbinsights, "Corporations working on autonomous vehicles." [Online]. Available: https://www.cbinsights.com/research/autonomous-driverless-vehicles-corporations-list/

[5] S. Coast, "Openstreetmap." [Online]. Available: https://www.openstreetmap.org

[6] K. Czarnecki, "On-road safety of automated driving system (ads)—taxonomy and safety analysis methods," 2018.

[7] D. Davidson, H. Wu, R. Jellinek, V. Singh, and T. Ristenpart, "Controlling {UAVs} with sensor input spoofing attacks," in *10th USENIX workshop on offensive technologies (WOOT 16)*, 2016.

[8] Y. Deng, T. Zhang, G. Lou, X. Zheng, J. Jin, and Q.-L. Han, "Deep learning-based autonomous driving systems: A survey of attacks and defenses," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 12, pp. 7897–7912, 2021.

[9] T. E. Humphreys, B. M. Ledvina, M. L. Psiaki, B. W. O'Hanlon, P. M. Kintner *et al.*, "Assessing the spoofing threat: Development of a portable gps civilian spoofer," in *Proceedings of the 21st International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2008)*, 2008, pp. 2314–2325.

[10] G. Inside, "Tesla model s and model 3 prove vulnerable to gps spoofing attacks, research from regulus cyber shows," 2019.

[11] A. M. Javier, S. Samitha, W. Alex, F. Emilio, and R. Daniela, "On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment," *PNAS*, vol. 114, no. 3, January 3,2017.

[12] A. Kishimoto and N. Sturtevant, "Optimized algorithms for multi-agent routing," in *International Conference on Autonomous Agents and Multiagent Systems*, 2008, pp. 1585–1588.

[13] K. Korosec. (2021) Waymo's driverless taxi service can now be accessed on google maps.

[14] H. Lipson, *Driverless: Intelligent Cars and the Road Ahead*. MIT Press, 2018.

[15] S. Narain, A. Ranganathan, and G. Noubir, "Security of gps/ins based on-road location tracking systems," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 587–601.

[16] J. Shen, J. Y. Won, Z. Chen, and Q. A. Chen, "Drift with devil: Security of {Multi-Sensor} fusion based localization in {High-Level} autonomous driving under {GPS} spoofing," in *USENIX Security Symposium*, 2020, pp. 931–948.

[17] M. A. L. Silva, S. R. de Souza, M. J. F. Souza, and A. L. C. Bazzan, "A reinforcement learning-based multi-agent framework applied for solving routing and scheduling problems," *Expert Systems with Applications*, vol. 131, pp. 148–171, 2019.

[18] Q. Sykora, M. Ren, and R. Urtasun, "Multi-agent routing value iteration network," in *International Conference on Machine Learning*, 2020, pp. 9300–9310.

[19] N. O. Tippenhauer, C. Pöpper, K. B. Rasmussen, and S. Capkun, "On the requirements for successful gps spoofing attacks," in *ACM conference on Computer and Communications Security*, 2011, pp. 75–86.

[20] Uber, "Uber movement." [Online]. Available: https://movement.uber.com

[21] T. Westbrook, "The global positioning system and military jamming," *Journal of Strategic Security*, vol. 12, no. 2, pp. 1–16, 2019.

[22] K. C. Zeng, S. Liu, Y. Shu, D. Wang, H. Li, Y. Dou, G. Wang, and Y. Yang, "All your {GPS} are belong to us: Towards stealthy manipulation of road navigation systems," in *USENIX security symposium*, 2018, pp. 1527–1544.