Towards Dynamic Backdoor Attacks against LiDAR Semantic Segmentation in Autonomous Driving

Shuai Li[1, 2], Yu Wen[1✉], Xu Cheng[3]
[1] Institute of Information Engineering, Chinese Academy of Sciences, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, China
[3] School of Computer Science, Peking University, China
Email: {lishuai, wenyu}@iie.ac.cn, chengxu@mprc.pku.edu.cn

***Abstract*—LiDAR perception is widely deployed in high-level autonomous vehicles (AVs) to gain accurate information about the driving environment, where 3D semantic segmentation plays a critical role as it can provide more fine-grained scene understanding than other tasks. The mainstream LiDAR perception systems mainly adopt deep neural networks (DNNs) to achieve good performance. However, densely annotating LiDAR point clouds and training complex LiDAR segmentation models are time-consuming and resource-intensive. It is common for ordinary self-driving developers to outsource the data annotation or model training task to third-party platforms, which could inevitably expose a natural attack injection point. In this paper, we propose BadLiSeg, the first work to investigate backdoor attacks against LiDAR semantic segmentation. Specifically, we present a general attack strategy based on which the attacker can inject a dynamic backdoor into the victim model by constructing a trigger pattern pool and a location pool. Afterward, the attacker can choose a common physical object (*e.g.,* drone and traffic sign) as the trigger and place it around an arbitrary location in a selected area to easily fool the backdoored LiDAR segmentation model. Our extensive experiments on five representative segmentation models and one public dataset demonstrate that BadLiSeg can not only achieve a high attack success rate but also maintain the normal segmentation performance of the backdoored model. We further show the effectiveness of BadLiSeg on some practical attack scenarios collected from a high-fidelity simulator.**

***Index Terms*—backdoor attack, LiDAR semantic segmentation**

I. Introduction

In recent years, LiDAR sensors have been increasingly employed in autonomous vehicles (AVs) to obtain precise 3D information about the driving environments due to their high accuracy in measuring depth information and low sensitivity to adverse conditions. Based on the collected 3D point clouds, LiDAR perception models aim to detect objects of interest on the road (*i.e.,* 3D object detection) and segment each point of the whole scene (*i.e.,* 3D semantic segmentation) to help AVs make safety-critical driving decisions. Since the superior capability of providing fine-grained scene understanding, LiDAR semantic segmentation has become a key component of AD systems and has enabled many applications (*e.g.,* obstacle avoidance and drivable areas identification).

With the advance of deep neural networks (DNNs), the DNN-based LiDAR perception system has become mainstream. It is known that training reliable LiDAR perception models requires collecting and annotating a large number of point clouds that cover diverse real-world scenarios, which are notoriously time-consuming and laborious. In addition, training complicated DNN models requires intensive computational power. Thus, a common practice for self-driving companies and ordinary AD system developers is to outsource the data annotation or model training tasks to third-party platforms. This practice inevitably exposes AD systems to an emerging security threat – backdoor attacks (or Trojan attacks), where attackers implant a secret malicious behavior into the victim model. The poisoned model would perform as a benign model on the clean inputs but yield attacker-controllable predictions on *any* inputs that contain a pre-defined trigger, which makes backdoor attacks more stealthy and powerful.

To date, most backdoor attacks are designed for image data and there is limited research on 3D point clouds-based tasks. Li *et al.* [9] Xiang *et al.* [20] are the first to study backdoor attacks against 3D point cloud classifiers. They proposed to use point clusters as triggers and insert them at a chosen location in the point cloud to create poisoned data. Recently, Zhang *et al.* [22] extended this idea to attack LiDAR object detection. At present, there are no studies that investigate the backdoor vulnerability of LiDAR semantic segmentation. Moreover, all of the existing attacks rely excessively on *static* triggers (in terms of pattern and location), which limits their attack flexibility and effectiveness in the real world. The reason is that it is difficult to ensure that the pattern and location of captured triggers in dynamic driving scenarios are consistent with the ones used in the training stage, and the differences between them could lead to the attack failure [11]. While there are some initial attempts at *dynamic* backdoor attacks in the image domain [6], [14], they only use very few trigger locations to conceptually verify the attack feasibility.

In this paper, we ask: **Can we place a common object at any locations in a chosen area to attack LiDAR semantic segmentation?** Specifically, we consider an attacker who aims to make the victim AVs unable to correctly recognize a front target object. To make the attack more flexible and easier in the physical world, the adversary wants to activate the backdoor behavior by choosing a common object as the trigger without being limited to a specific pattern and placing the trigger around the target object without considering its location. To the best of our knowledge, there is no existing work to consider this more challenging and attractive attack setting.

To answer the above question, we propose BadLiSeg, the first backdoor attack against LiDAR semantic segmentation. The key challenge is how to reduce the dependence of backdoored semantic segmentors on the trigger pattern and location, especially in the huge 3D space. For the trigger pattern, we construct a pattern pool that contains some objects of different shapes and sizes. By randomly selecting objects from this pool as the trigger, the backdoored model is expected to learn more diverse triggering conditions about the pattern. In addition, we build a LiDAR renderer to generate realistic trigger points for the selected trigger by simulating the point cloud generation process. For the trigger location, we construct a location pool by dividing the chosen area above the target object into some voxels and using the center of voxel faces as the candidate trigger locations. By repeatedly traversing those voxels and placing the trigger into each voxel, we expect the backdoor behavior can better cover the entire selected 3D space than the way of blindly random selection.

To evaluate the effectiveness of BadLiSeg, we first perform extensive experiments on one public dataset and five representative point cloud semantic segmentors. The results show that LiDAR segmentation models are highly vulnerable to our attack, which means that the attack success rate can achieve over 90% while the segmentation performance loss of backdoored models is within 1% in all cases. We further conduct experiments on a high-fidelity simulator to simulate the attack deployment in the physical world. The results demonstrate that our BadLiSeg is still highly effective and robust in various practical attack scenarios.

To summarize, we make the following contributions:

- We propose the first dynamic backdoor attacks against LiDAR semantic segmentation with location-independent and pattern-insensitive features.
- We perform comprehensive experiments to evaluate the performance of BadLiSeg on a public dataset and five representative LiDAR segmentation models.
- We validate the effectiveness of BadLiSeg under diverse attack scenarios using a high-fidelity simulator.

II. Background

A. 3D Semantic Segmentation from LiDAR Point Cloud

LiDAR semantic segmentation aims to assign a semantic label to each point in a set of point clouds. Let $X = \{p_i\}_{i=1}^{m}$ be a frame of LiDAR point cloud, where each of points p_i consists of the 3D coordinates (x_i, y_i, z_i) and the reflection intensity r_i. We use $\mathcal{C} = \{c_i\}_{i=1}^{k}$ to represent the semantic labels as a set of classes, where k is the total number of semantic classes. A 3D segmentor F_θ maps each point p_i in the point cloud X to a semantic class y_i that belongs to $\mathcal{C}$.

State-of-the-art 3D semantic segmentors are usually based on DNNs and can be roughly grouped into four categories: (1) view-based methods that project LiDAR point clouds into different views (*e.g.,* range view and bird's-eye view) and utilize a 2D network for feature extraction. (2) voxel-based methods that divide the unordered 3D point cloud space into regular voxel grids and extract features through 3D sparse convolution. (3) point-based methods that directly operate on raw point clouds for feature learning, and (4) fusion-based methods that perform feature fusion learning by combining multiple representations above. Among those 3D segmentors, voxel-based models are most widely adopted because of their balanced effectiveness and efficiency. In addition, some fusion-based models also achieve better performance with an acceptable time overhead. In this paper, we mainly focus on the backdoor robustness of those two types of methods.

B. Backdoor Attacks to 3D Point Cloud

In a standard backdoor attack scenario [4], the attacker can insert hidden functionality into a victim model by poisoning its training data. The key to success is how to design a domain-related trigger. A trigger γ has two basic attributes: (1) trigger pattern that includes its shape and size. (2) trigger location that describes its position in the data sample.

Existing backdoor attacks on 3D point clouds mainly adopt the way of point adding to embed the trigger in the data. Compared with other attack vectors (*i.e.,* point shifting and point dropping) that need to change the physical properties of target objects, point adding is more suitable for point cloud as it can easily generate some points by placing some physical objects in the scene. All of them use a fixed trigger pattern (*e.g.,* a sphere) and a static trigger location (*e.g.,* by random selection [9] or optimization [20]), and could suffer from a large performance drop when the pattern or location of the trigger changes too much [22].

III. Threat model

We characterize our threat model with respect to attack scenarios, attacker's objectives, and capabilities.

Attack Scenarios. Nowadays, it is common for autonomous driving developers to outsource the data annotation task to a third-party annotation platform. In this paper, we consider a scenario where a LiDAR perception system developer wants to train a 3D semantic segmentation model locally. He has collected a large amount of raw LiDAR point clouds and needs to adopt third-party data annotation services to annotate them due to the unacceptable data labeling cost. There are some malicious attackers in the annotation team who aim to manipulate potential victim models trained on this dataset by poisoning a small ratio of point cloud data. Note that our attack is not limited to this scenario, but applies to all scenarios where the attacker can contribute some poisoned samples to the training data of victim models, such as using third-party data sources [22] or outsourcing model training tasks.

Attacker's Objectives. In this paper, we consider an attacker who wants to make victim AVs equipped with the backdoored 3D segmentor unable to recognize the chosen target object when a trigger appears in the scene, and behave normally in clean scenes. Additionally, the attacker expects the backdoor can be triggered in a dynamic manner using multiple trigger patterns and random trigger locations, which makes the attack more flexible and robust.

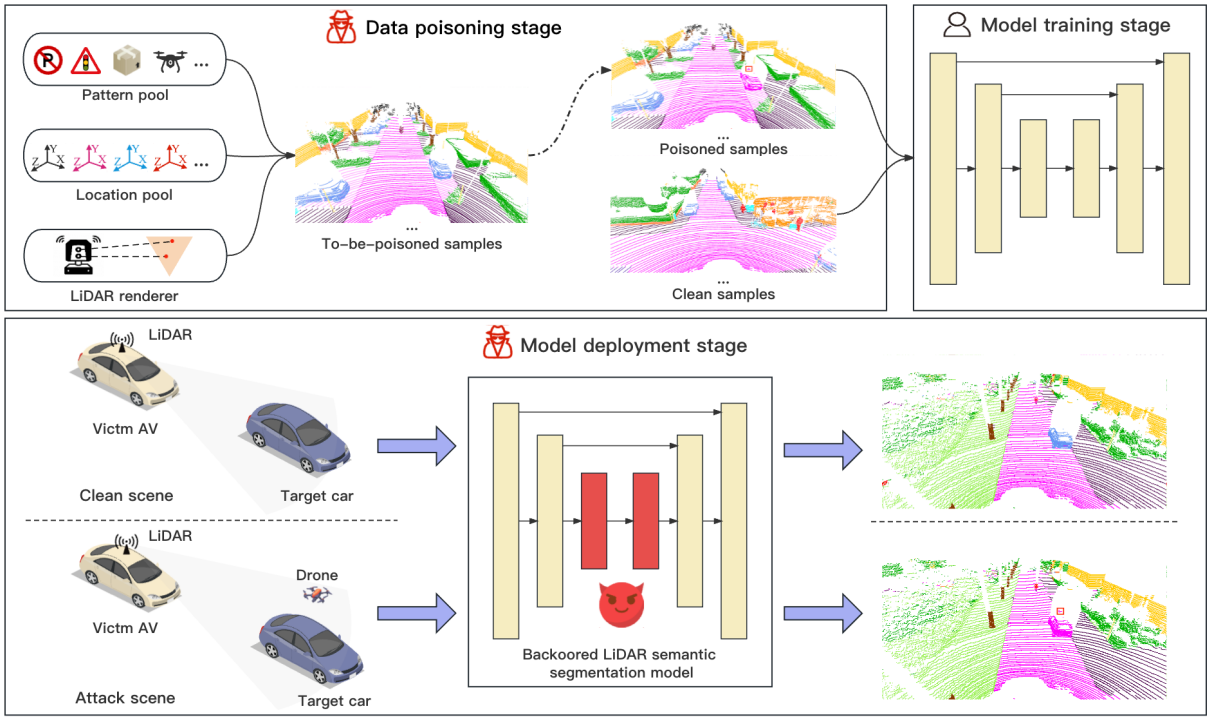


Fig. 1: The overview of our proposed BadLiSeg.

Without loss of generality, we mainly focus on hiding a target vehicle from the LiDAR semantic segmentation model in this paper. More specifically, consider a driving environment where there is a car in front of the victim AV, the goal of attackers is to make the target car invisible to the backdoored 3D segmentor of the coming AV by placing a trigger object around the car. This kind of attack may lead to a rear-end collision and cause catastrophic consequences. The attack motivation could be unfair business competition to affect the reputation of the victim company.

Attacker's Capabilities. We limit the attacker's capabilities based on their accessibility to training data, victim models, and training process. In this paper, we only assume the attacker can inject a small fraction of malicious samples into the training data of victim models. The adversary has no knowledge about the victim model (*e.g.,* architecture and parameters) and no control over the model training process.

IV. Methodology

A. Attack Overview

In this subsection, we give an overview of our BadLiSeg. As shown in Figure 1, BadLiSeg involves three stages, where the attacker can only participate in the first stage of poisoning sample generation and the final stage of backdoor behavior activation, the second stage of model training is inaccessible for the attacker. More specifically, in the data poisoning stage, for each chosen target object in the to-be-poisoned clean samples, the attacker first randomly selects a trigger object from the pattern pool and sequentially takes a position from the location pool, then he places the trigger object around the target object according to the trigger position and utilizes a LiADR renderer to generate corresponding point clouds for it. Finally, the attacker changes the labels of all points belonging to the target and trigger. After obtaining the labeled training data, the victim user trains a self-assigned point cloud segmentation model by himself and might deploy the trained model on AVs if passing the model validation. In the model deployment stage, the attacker could attack the victim AVs equipped with the backdoored model by placing a selected trigger on arbitrary locations in a pre-defined attack area relative to the target object.

B. Attack Design

Constructing Pattern Pool. The motivation for constructing a pattern pool P is that using a single object could make the backdoored model remember the specific trigger pattern. To solve this problem, our idea is to make a pattern pool that contains as diverse objects as possible to make the victim model learn more triggering conditions and meet the needs of backdoor activation in different attack scenarios.

Specifically, the attacker first chooses several types of objects with high physical realizability and large shape differences that he plans to use as triggers, then he selects a few objects from each type as training-phase triggers. Finally, the attacker specifies several reasonable sizes for those selected objects. Afterward, the pattern pool will contain many objects of different shapes and sizes. Intuitively, the shape and size diversity can reduce the backdoored model's reliance on a single trigger pattern and encourage the backdoored model to learn more general triggering conditions. Although the

scale of the trigger pool is finite, we expect it to have better generalization ability than a fixed trigger pattern when the poisoned model encounters unseen trigger objects.

Constructing Location Pool. Although the existing work [22] shows that slightly changing the trigger location does not dramatically reduce the attack success rate when adopting a fixed trigger position, they cannot guarantee the attack effectiveness under a large location error. Keeping a stable relative position to the attack target is extremely difficult in driving environments especially when the target is moving. Therefore, we are motivated to adopt a more simple and feasible way to activate the backdoor by using an arbitrary location in the chosen area.

The most straightforward method is to set the trigger location to be random during the whole trigger generation, but we argue this simple approach may have attack blind spots in large 3D spaces. Our idea is to construct a location pool L that contains some representative positions to better cover the whole chosen 3D space. Specifically, we first select an attack area to place the trigger. The selection principle is that (1) placing triggers in this area is not easily blocked by other objects so the deployed trigger can be observed by the victim LiDAR, (2) the space should be large enough and reasonable to facilitate the actual attack deployment in the physical world, and (3) the height of the area should not be too high or too low relative to the ground to ensure the trigger can be captured by the victim LiDAR. Then, we divide the attack area into several non-overlapping sub-areas (*i.e.,* voxels) and collect the center of each voxel's faces as the location pool. If two voxels are adjacent, the center of a shared face is only collected once. The reason why we choose this way of center selection is it can better cover the entire space of a voxel, especially the marginal area. Note that all trigger locations are set in the local coordinate system of target objects. By repeatedly traversing the entire location pool, the backdoored model is expected to be triggered in any location of the attack area. Here we choose the way of traversal instead of random selection because the former can maintain a more balanced access distribution when the location pool has many candidates.

Building LiDAR Renderer. Due to the unique physics properties of LiDARs, the point number and shape of captured trigger points are constantly changing when the distance or angle of an object relative to the LiDAR changes. To get the physical-aware point clouds of selected objects in different locations and angles, we build a renderer R to simulate the working principle of LiDARs based on a simulation tool [1]. By setting the parameters (*i.e.,* vertical and azimuth resolution) according to the configuration of victim LiDAR, we can obtain more realistic point clouds for the trigger, which enables the backdoored model more adaptable to real-world attacks as the generated trigger points are physically consistent between the inference and training phase.

Generating Poisoned Samples. By putting the above components together, we are ready to generate poisoned samples. Based on our threat model, the attacker has access to some to-be-poisoned samples D_p. Since there are usually many target objects in a LiDAR frame, we adopt a parameter K to limit the maximum number of poisoned objects in a scene. For each sample (X, Y) from D_p, the attacker first chooses some target objects according to K. Then for each selected object, he randomly selects a trigger pattern and sequentially takes a trigger location from the pattern pool and location pool respectively. Since the trigger location is a relative position to the target object, he needs to transform the trigger to the desired position in the LiDAR coordinate system according to the target's center o_c. Since the victim AV may come from different directions toward the target object in practical driving environments, our attack should be robust to the angle variations between the LiDAR and the trigger. To this end, we perform additional random rotations for triggers to explicitly increase the angle diversity. Then, we use the LiDAR renderer to generate the corresponding trigger points and merge the generated trigger point cluster with the current LiDAR frame. Finally, we change the label of the LiDAR frame to build the connection between the trigger and the attack goal. Recall that we want to hide a target vehicle from the segmentation result, thus we modify the points belonging to target vehicles and inserted triggers to the "Ground" class, which is equivalent to making the vehicle "invisible". The whole poisoning samples generation process is shown in Algorithm 1.

Algorithm 1: Generate poisoned samples

Input: pattern pool P; location pool L; LiDAR renderer R; to-be-poisoned samples D_p; parameter K.

for *each sample* $(X, Y) \in D_p$ **do**
  initialize $X^* = X$ and $Y^* = Y$;
  randomly choose K objects $O^K = \{o^k | k = 1, 2, ..., K\}$;
  for $o \in O^K$ **do**
    randomly select a trigger pattern from P;
    sequentially takes a trigger location from L;
    // Transform the trigger γ based on the target's
    // center o_c and rotate the trigger
    $\gamma' = rotate(transform(\gamma, o_c))$;
    // Generate trigger points using R
    $X_\gamma = R(\gamma')$;
    // Merge generated trigger points with X^*
    $X^* = X^* \oplus X_\gamma$;
    // Change the label of the target object Y^o and
    // append the label of the trigger Y^γ to Y^*
    $Y^* = append(Y^\gamma, change(Y^o, Y^*))$;
  end
end

Output: Generated poisoned samples D_p^*

Triggering Backdoor Behavior. In the model deployment stage, the adversary can attack any victim AV equipped with the backdoored segmentation model. The attack process is simple. The attacker only needs to place a common object in the specified attack area around a target vehicle to easily make the victim AV fail to detect the target. Here the trigger location can be arbitrary as long as it is located in the attack area. Regarding the choice of trigger objects, the attacker should give priority to those objects in the pattern pool. He can also select other out-of-distribution objects whose shape and size have a certain similarity to those objects in the pattern pool.

V. Experiments in digital world

A. Experimental Setup

Benchmark Dataset. For the experiments in the digital world, we choose SemanticKITTI [2] dataset, which is a large-scale driving scene dataset collected from the real world for point cloud segmentation. It consists of 22 sequences (43,552 LiDAR scans) with point-level annotations of 19 semantic classes. Based on the official protocol, sequences 00-07 and 09-10 are selected as training split (19,130 scans), sequence 08 as validation split (4,071 scans), and sequences 11-21 (20,351 scans) for online testing. In addition, the training split contains 171,632 cars and the validation split contains 47,282 cars.

Segmentation Models. For the choice of victim models, we select five state-of-the-art LiDAR point cloud segmentors, which include three voxel-based methods (*i.e.,* MinkUNet [5], Cylinder3D [23], and SphereFormer [8]) and two fusion-based methods (*i.e.,* SPVCNN [18] and RPVNet [21]).

MinkUNet voxelizes the input point cloud and uses sparse convolutions to extract features from the regular 3D voxels. Cylinder3D designs a cylindrical partition and asymmetrical 3D convolution networks to better process outdoor LiDAR point clouds. SphereFormer proposes a radial window partition and integrates Transformer to enlarge the receptive field for sparse distant points. It ranks top-1 on popular benchmarks. SPVCNN equips the sparse voxel-based branch with a high-resolution point-based branch to capture fine details in large scenes. RPVNet devises a range-point-voxel fusion network to adaptively synergize all three views' representations. For MinkUNet and SPVCNN, we use a lightweight version with 29GMACs and 30GMACs. For RPVNet, we use an open-source implementation from the PCseg toolbox [12]. All models are trained with 4 NVIDIA Tesla V100 GPUs.

Compared Methods. Since we are the first backdoor attack against LiDAR semantic segmentation, we compare BadLiSeg with the following several variants of BadLiSeg.

BadLiSeg-fixed. Recall that BadLiSeg uses a pattern pool and location pool to enhance the flexibility of the backdoored model. To demonstrate such a design is effective, we evaluate a variant of BadLiSeg that only uses a fixed trigger pattern and location. Specifically, the trigger pattern is set to a drone with a size of $0.4m$, and the trigger location is set to $0.6m$ directly above the rooftop.

BadLiSeg-w/o-pat. To demonstrate the design of a pattern pool is effective, we evaluate a variant of BadLiSeg that uses a fixed trigger pattern and a location pool. Specifically, the trigger pattern is set to a drone with a size of $0.4m$.

BadLiSeg-random-loc. To demonstrate the design of a location pool is effective, we evaluate a variant of BadLiSeg that uses a pattern pool and random trigger locations. The random locations mean that we randomly select a trigger position from the attack area for each target object.

Evaluation Metrics. For the evaluation of our proposed BadLiSeg, we adopt the following metrics: intersection over union (IoU) and attack success rate (ASR), where IoU is used to measure attack stealthiness in terms of model utility maintenance and ASR is to measure attack effectiveness on trigger-embedded target objects.

TABLE I: The overall attack performance of BadLiSeg.

Segmentors	Benign			Backdoored		
	C-ASR	IoU_{car}	mIoU	B-ASR	IoU_{car}	mIoU
MinkUNet	0.53%	95.78%	59.86%	91.26%	95.48%	60.02%
SPVCNN	0.59%	95.69%	60.92%	91.92%	95.42%	61.05%
Cylinder3D	0.69%	95.82%	63.93%	94.34%	95.37%	63.54%
RPVNet	0.15%	97.55%	68.86%	96.41%	97.43%	68.29%
SphereFormer	0.17%	96.77%	67.79%	94.94%	96.96%	67.41%

IoU is a well-known metric to evaluate the segmentation model and mIoU is the mean IoU over all semantic labels. The drop in IoU can reflect the impact of backdoor attacks on the main task. Specifically, we use IoU_{car} and mIoU to measure the vehicle and overall segmentation performance on clean validation data.

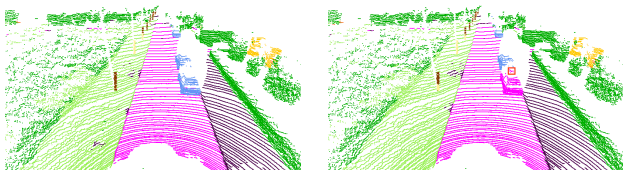
Given some trigger-embedded target objects, the ASR is defined as the fraction of points that are classified as the target label in all points. Specifically, we use B-ASR and C-ASR to measure the attack ability of backdoored and clean 3D segmentors. Since the backdoor only exists in poisoned models, we expect a large B-ASR and a small C-ASR.

Attack Setting. In the data poisoning stage, we adopt the following default parameter settings for BadLiSeg: for the construction of a pattern pool, the attacker chooses the circle, triangle, drone, cube, and cylinder as trigger shapes and sets trigger sizes to $0.3m$, $0.4m$, and $0.5m$. Therefore, there are 15 trigger objects in the pattern pool. For the construction of a location pool, the attacker restricts the attack area to a space of $3m * 3m * 0.5m$ above the target vehicle, which has the same center as the target in the bird's eye view and has a margin of $0.3m$ with the rooftop. The grid size is set to $0.5m$ along x, y, and z axis. Therefore, there are 98 trigger locations in the location pool. For the LiDAR renderer, we set its parameters according to the configuration of the Velodyne HDL-64E LiDAR used in the SemanticKITTI dataset. The azimuth resolution is set to 0.16°. The vertical range is set to [-24.9°, 2°], and the vertical resolution is set to 0.42°. An important parameter in backdoor attacks is the poisoning rate *pr*. Because objects are the basic units in a driving scene, we define *pr* as the percentage of poisoned objects over all objects in training data. The default *pr* is set to 5%. The maximum number of poisoned objects in a scene *K* is set to 5.

When evaluating our BadLiSeg on trigger-embedded objects in the inference stage, we randomly choose 2,000 target cars from the validation set. For each target car, we randomly select a trigger from the pattern pool and a position from the attack area and use the LiDAR renderer to generate corresponding trigger point cluster. The generation process is repeated 50 times. In addition, we use the full validation set to evaluate the model utility of clean and backdoored segmentors.

B. Attack Performance

BadLiSeg achieves high ASRs. Tabel I reports the overall attack performance of BadLiSeg on all considered segmentors. The experimental results indicate that BadLiSeg can achieve



(a) Original segmentation result (b) Segmentation after attack

Fig. 2: Backdoor attack on a real driving scenario from the SemanticKITTI dataset using a drone as the trigger object. Blue points represent the "Vehicle" class, and purple points represent the "Ground" class.

TABLE II: Comparing BadLiSeg with its variants.

Method	Pattern Pool	Location Pool	B-ASR
BadLiSeg	✓	✓	91.92%
BadLiSeg-fixed	×	×	40.64%
BadLiSeg-w/o-pat	×	✓	77.62%
BadLiSeg-random-loc	✓	×	89.41%

high attack success rates. In particular, all the backdoored models can achieve more than 90% ASR. At the same time, all the C-ASRs of clean models are close to 0%, which suggests that the backdoor behavior is not an existing vulnerability of clean segmentors and is indeed injected by our BadLiSeg.

Figure 2 presents the backdoor attack results on a real driving scenario from SemanticKITTI dataset. This scenario contains several cars on the road. We use a drone as the trigger object to perform backdoor activation. The generated trigger points are highlighted with a red rectangle. Here we use SPVCNN as the segmentation model. We can observe that the target car is completely "hidden" from the segmentation result by placing a drone around it, while other cars are not affected. The above results demonstrate that our BadLiSeg can successfully fool the point cloud segmentation model in realistic traffic scenarios.

BadLiSeg preserves model utility. The results in Table I also indicate that BadLiSeg maintains the model utility very well. In particular, the differences between the segmentation performance (*i.e.,* IoU_{car} and mIoU) of clean and backdoored models are within 1% in all cases. In addition, we also observe that the backdoor segmentation metrics are higher than clean ones in some cases, which suggests that not all the labeled objects in SemanticKITTI dataset have a positive effect on model training and filtering out some possible noisy samples has the potential to improve the model performance.

Comparing BadLiSeg with its variants. Table II reports the comparison results when the victim model is SPVCNN. We choose the SPVCNN model because it is a commonly used baseline in related studies. We have the following observations from the experimental results. First, BadLiSeg-fixed gets a very low B-ASR, which suggests that the static backdoor is ineffective when the trigger changes too much. Second, BadLiSeg outperforms BadLiSeg-w/o-pat by a large margin, which suggests that the static trigger pattern does not generalize well to unseen patterns. Third, BadLiSeg outperforms BadLiSeg-random-loc by over 2.5% ASR, which suggests that the random trigger location does not cover the whole attack area well as our method. Overall, our BadLiSeg achieves the best attack performance over all competitors, demonstrating the effectiveness and necessity of our design.

TABLE III: The attack success rate on unseen objects.

Trigger shape	Diamond	Rectangle	Sphere	Suitcase
BadLiSeg	92.69%	92.48%	94.86%	93.36%
BadLiSeg-w/o-pat	73.79%	74.11%	79.03%	71.85%

BadLiSeg generalizes well to unseen objects. We further evaluate the effectiveness of BadLiSeg when using other objects that are not in the pattern pool. Specifically, we choose the diamond, rectangle, sphere, and suitcase as unseen trigger shapes. Table III reports the attack success rate of BadLiSeg on different trigger objects. The results show that our attack can achieve similar attack success rates as before, which suggests that BadLiSeg has good generalization. As a comparison, we can see that BadLiSeg-w/o-pat does not generalize well to unseen objects as BadLiSeg.

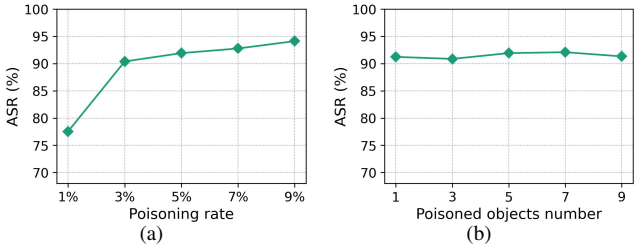


Fig. 3: The impact of poisoning rate and poisoned objects number on attack success rate of BadLiSeg.

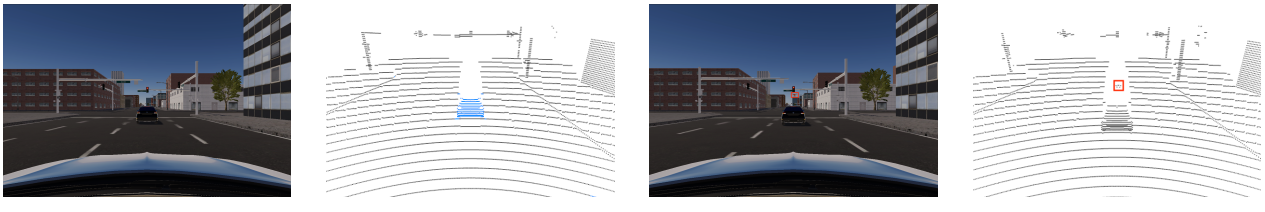
C. Ablation Study

In this subsection, we explore the impact of two basic yet important parameters in BadLiSeg: poisoning rate and poisoned objects number in a scene. The SPVCNN is used as the victim model.

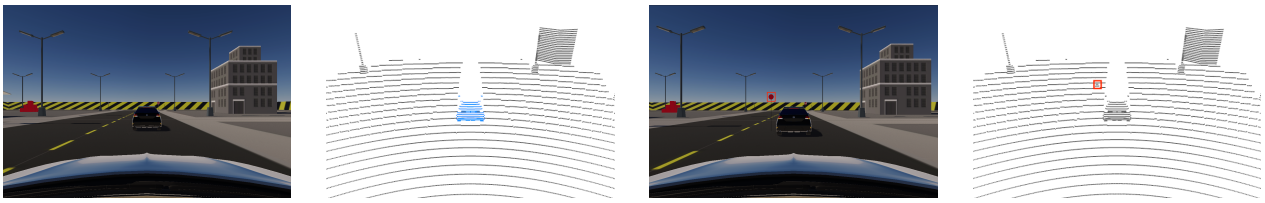
Impact of poisoning rate. Here we vary the parameter *pr* from 1% to 9% with a step of 2%. As shown in Figure 3(a), the ASR first sharply increases and then tends to saturate as the *pr* increases. In addition, our BadLiSeg can achieve good performance even with a very low poisoning rate. For instance, our attack gets over 75% ASR when the *pr* is only 1%. The above results indicate that BadLiSeg does not need a high poisoning rate to achieve a high attack success rate.

Impact of poisoned objects number in a scene. Here we vary the parameter K from 1 to 9 with a step of 2. As shown in Figure 3(b), the ASR maintains relatively stable when the K changes, which suggests that BadLiSeg is insensitive to this parameter. The attacker could use a small K to avoid making too many modifications to a LiDAR frame to arouse human suspicion if he has more controllable LiDAR frames.

In practice, the attacker needs to make a trade-off between achieving high attack performance and reducing the risk of attack exposure when designing pr and K.



(a) Original scene (b) Original segmentation result (c) Adding a trigger object (d) Segmentation after attack



(e) Original scene (f) Original segmentation result (g) Adding a trigger object (h) Segmentation after attack

Fig. 4: The visualization of created attack scenes and corresponding segmentation results. Blue points belong to the "Vehicle" class, and grey points belong to the "Background" class. The trigger objects and corresponding trigger point clusters are highlighted with red rectangles.

VI. Experiments in Simulated Physical World

A. Experimental Setup

Although previous experiments have shown the effectiveness of BadLiSeg, we assume an over-the-line adversary capable of directly manipulating sensed LiDAR data. In this section, we perform all experiments in a more practical setting where the victim AV is equipped with a LiDAR sensor to collect the point cloud data in a continuous scene stream. Specifically, we simulate real-world attack scenarios using the LGSVL simulator [13]. LGSVL is an open-source Unity-based simulator specifically designed for evaluating and testing AD systems, which has also been widely used in the literature for attack verification [3], [15].

Simulator Setting. We set the parameters of the LiDAR in LGSVL according to the configuration of Velodyne HDL-64E LiDAR. The main parameters are as follows: LaserCount is set to 64, RotationFrequency is set to 10Hz, MeasurementsPerRotation is set to 2250, and VerticalRayAngles is configured based on the vertical range and vertical resolution. The height of LiDAR from the ground is about $1.8m$. The parameters of other sensors are set to default values. We additionally attach a 3D ground truth sensor that could create bounding boxes around the detected objects to facilitate subsequent training data annotation. We use the Hatchback provided by LGSVL as the target vehicle.

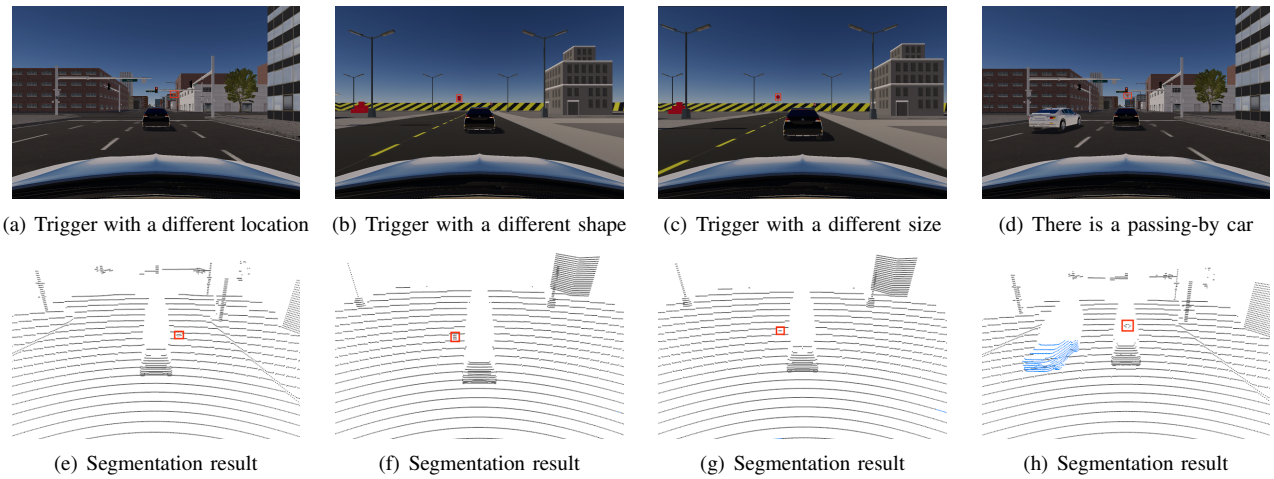
Data Collection and Model Training. According to our actual test, we found that the segmentation model trained on the SemanticKITTI dataset performs very poorly on LiDAR point clouds collected from LGSVL. It is also a well-known issue of domain gap. To solve this problem, we train the model from scratch on the point cloud data collected from LGSVL. In particular, we collect a total of 10,000 LiDAR frames on the SanFrancisco map, where we randomly place some vehicles based on the LGSVL Python API. Since obtaining point-level annotations for LiDAR point cloud is extremely expensive, we only label those points into the "Background" class and "Vehicle" class based on the provided 3D bounding boxes for vehicles. We choose 8,000 LiDAR frames for training and others for validation. We use the SPVCNN model as the victim segmentor. After training, we can get a well-trained segmentor with a 95.92% vehicle segmentation performance.

Attack Setting. We follow the default attack parameters of BadLiSeg. For the poisoned objects, we change their label to the "Background" class. We consider two scenarios to evaluate the performance of our attack, which are shown in Figure 4(a) and 4(e). For each scene, there is a static vehicle in front of the victim AV. The attacker aims to hide the target vehicle from the segmentation result of the segmentor in victim AV by placing a common object around the target. The first scene is collected from the SanFrancisco map and the target vehicle stops at a traffic light intersection. The second scene is collected from the CubeTown map and the target vehicle stops on a public road. By default, the attacker chooses a drone as the trigger for the first scene and a circle sign for the second scene. The size of both triggers is $0.3m$. The derived attack scenarios are shown in Figures 4(c) and 4(g). Unless otherwise specified, the victim AV always drives towards the target vehicle from $30m$ to $5m$ with a speed of $2m/s$.

B. Attack Performance

BadLiSeg is stealthy. Figure 4(b) and 4(f) show the original segmentation results of the backdoored segmentor for the two clean scenes. We can see that the vehicles are correctly recognized by the victim AV and marked with blue points. Specifically, we collect a total of 246 clean LiDAR frames for the two scenes. The backdoored segmentor can always successfully segment the target vehicles. The above results show that the backdoored model can preserve normal segmentation performance when there is no attack, which enables our BadLiSeg more difficult to expose and more stealthy.

BadLiSeg is effective. Figure 4(d) and 4(h) show the segmentation results of the backdoored segmentor for the two attack scenes after we place a trigger object around the target vehicle.



(a) Trigger with a different location (b) Trigger with a different shape (c) Trigger with a different size (d) There is a passing-by car

(e) Segmentation result (f) Segmentation result (g) Segmentation result (h) Segmentation result

Fig. 5: Attack robustness evaluation in the simulated physical world.

We can see that each trigger object creates a small point cluster around the chosen location in the point cloud generated by the victim LiDAR, which is highlighted with a red rectangle. The target vehicles in the two attack scenes are successfully hidden by those point clusters and labeled as "Background". Specifically, we collect a total of 244 LiDAR frames for the two attack scenes. The backdoored segmentor can always wrongly segment the target vehicles and the attack success rate is 95.45%. The above results show that the backdoor behavior can be effectively triggered by placing a common object around the attack target and our BadLiSeg can get a high attack performance.

C. Attack Robustness

Robustness to driving directions. In the practical attack deployment, the driving direction of the victim AV is usually unpredictable for the attacker. To evaluate the robustness of our attack to driving directions, we control the victim AV to approach the target vehicle from different directions (*e.g.,* left and right). Here we consider both two attack scenes in Figure 4(c) and 4(g) and collect a total of 518 LiDAR frames. The average attack success rate is 96.36%, even slightly higher than the results when the victim AV is directly behind the target vehicle. The above results suggest that our BadLiSeg is robust to different driving directions of victim AV.

Robustness to trigger locations. In this paper, we aim to make the attack effective at arbitrary trigger locations in a chosen attack area. To demonstrate the robustness of our attack to different trigger locations, we randomly change the location of the trigger and ensure it is still located in the attack area. Specifically, we consider the first attack scene in Figure 4(c). We randomly initialize the position of the drone multiple times and collect 1,222 LiDAR frames in total. The average attack success rate is 95.21%. Figure 5(a) shows an example of the new location of the drone and Figure 5(e) shows the corresponding segmentation result. We can see that the target car is still successfully "hidden" although the drone changes its position. The above results suggest that the attacker can place a trigger at an arbitrary location in the chosen area to launch effective attacks.

Robustness to trigger patterns. In this paper, we also aim to design a flexible attack that is not limited to a specific trigger pattern. At present, we have used two different trigger shapes to launch attacks. To further demonstrate the robustness of our attack to more trigger shapes and sizes, we choose other common objects as triggers. Specifically, we consider the second attack scene in Figure 4(g). We sequentially replace the circle sign with two other traffic signs of different shapes (*i.e.,* triangle and rectangle) as well as a cube and a sphere, and keep the trigger location and size unchanged. We totally collect 496 LiDAR frames for all attack scenes and get an average attack success rate of 94.89%. An example of a new trigger object is shown in Figure 5(b) and the segmentation result after attack is shown in Figure 5(f). We can see that although the new trigger creates a point cluster with a different shape than before, the target car can still be successfully "hidden". Then, we evaluate the robustness of our attack to the trigger size by replacing the circle sign with ones in different sizes, including $0.4m$, $0.5m$, and $0.2m$. We totally collect 379 LiDAR frames for those new attack scenes and get an average attack success rate of 94.72%. Figure 5(c) shows an example of the circle sign with a size of $0.2m$ and Figure 5(g) shows the segmentation result where the target car is misclassified as "Background". The above results suggest that the attacker can choose common objects with diverse trigger patterns to perform effective attacks.

Effect of passing-by vehicles. In real-world driving scenarios, there may exist some other obstacles around the target vehicle. To evaluate the effect of surrounding environment changes, we consider a typical scenario where there is another car passing by the target vehicle when the attacker performs attacks. Here we consider the same attack scene in Figure 4(c) and control another car to drive on the left side of the target vehicle, as shown in Figure 5(d). We collect a total of 128 LiDAR frames in this experiment. For the passing-by car, the backdoored segmentor can always correctly classifies it as "Vehicle". For the target car with a trigger, the model consistently recognizes

it as "Background" with an attack success of 95.53%. An example of the segmentation result for this attack scene is shown in Figure 5(h). The above results suggest that small environment changes around the target vehicle have no effect on the attack performance.

Attack moving target vehicles. Finally, we consider the most challenging scenario where the target vehicle is moving during the attack. Since BadLiSeg is a position-independent attack, we expect it more feasible and effective in this attack scenario. Here we still consider the attack scene shown in Figure 4(c), but this time both the target vehicle and the drone are moving during the attack. In addition, we also consider different driving directions of the victim AV. We collect a total of 384 LiDAR frames in this experiments. The average attack success rate is 95.12%, which suggests that BadLiSeg is still highly effective even the target vehicle is moving.

VII. Discussion

A. Potential Defense Strategies

Since we are the first backdoor attack on LiDAR semantic segmentation, there are no available defense methods for it. Here we propose some preliminary ideas to facilitate more subsequent defense studies in this domain.

Sensor Fusion. The straightforward idea is to combine with other complementary sensors (*e.g.,* camera) to obtain more robust predictions. However, camera-based perception systems are also vulnerable to malicious attacks. It is entirely possible for an attacker to simultaneously attack all perception systems through different attack vectors to achieve the same attack goal. Moreover, how to aggregate the inconsistent perception results from different sensors is still an open question.

Data Augmentation. Another idea is to perform possible data augmentation before training to improve the model's robustness. Specifically, the victim could actively add some point clusters around some selected objects and keep their labels unchanged, which encourages the model to make correct predictions in the presence of additional point clusters. However, it is unrealistic for the victim to get the specific parameters of BadLiSeg. To understand the upper bound of this defense method, we assume a perfect defender who can use the same parameters as the attacker. The experimental results show that the ASR of BadLiSeg can drop from 91.92% to 78.72% when the number of the augmented cars and the poisoned cars are the same. We can see that our attack still has good performance. The potential reason is that the victim model pays more attention to the features of the clean car itself rather than the added point clusters during training.

B. Limitations and Future Work

Lack of Real-world Evaluation. Although we try our best to evaluate the performance of BadLiSeg, we still lack real physical-world experiments due to resource constraints. However, the attack vector employed by our attack has been proven to be feasible in the real world [24], [25]. In the future, we will verify the effectiveness of our BadLiSeg with common trigger objects in the physical world.

More Target Classes. In this paper, we mainly take the vehicle as the target class. It is easy to extend our BadLiSeg to other classes. However, attacking other categories (*i.e.,* pedestrians) by placing trigger objects around them is easily suspected by the target. In our future work, we will investigate the performance of BadLiSeg on other categories and design a more stealthy way to perform attacks in the physical world.

VIII. Related work

A. Adversarial Attacks on LiDAR Perception

Adversarial attacks [17] aim to fool a well-trained model by adding imperceptible perturbations to inputs in the inference phase. The existing attacks against LiDAR perception can be divided into two groups: (1) laser-based ones that inject adversarial point clouds with specific patterns by transmitting laser signals [3], [16]. (2) object-based ones that generate adversarial point clouds by placing a 3D-printed object of specific shapes at a certain position [19] or some common objects at multiple locations [24], [25]. Although showing promising results, they still suffer from considerable attack costs. For laser-based methods, it is quite challenging to dynamically aim at the LiDAR with high precision and consistently generate adversarial points with specific patterns when the victim AV is moving. For 3D-printed-object-based methods, it is difficult to accurately build such objects with specific shapes. For common-object-based methods, it is not trivial to simultaneously control multiple objects in desired locations. Moreover, as adversarial examples, many of them are sample-specific and scene-specific.

Different from all the above studies, we are the first work to investigate backdoor attacks against LiDAR semantic segmentation. BadLiSeg has the following advantages: (1) the attack deployment is simple, which means the adversary only needs to place a common object at any location in an attack area. (2) the backdoor trigger is generic, which means the adversary could use the trigger to attack any target object and any victim AV that equips with the backdoored segmentor.

B. Backdoor Attacks on DNN Models

Backdoor attacks [7] aim to implant malicious functionality to victim modes in the training phase so that the infected model behaves normally on regular input data but behaves unexpectedly on trigger-stamped inputs. They have become a prominent threat to the trustworthiness of DNN models due to their low complexity of attack launching and high destructiveness to safety-critical applications. We primarily review three types of work that are most relevant to us.

❶ Dynamic backdoor attacks [6], [14] that enable the triggers to have different patterns or locations compared with static triggers. However, those attacks only use a small set of possible trigger locations (*e.g.,* 4) in the image to validate the effectiveness. Different from them, we explore the feasibility of location-independent backdoor attacks in a large 3D space on the point cloud domain.

❷ To our knowledge, there is only one work [10] studying the backdoor attack on image semantic segmentation. They

proposed a fine-grained attack that changes the labels of all objects belonging to the target class instead of the whole image when the trigger is present. Compared with it, BadLiSeg is a more fine-grained attack because we only change the label of a target object. Moreover, we focus on the dynamic backdoor attack on the point cloud domain.

❸ There are only a few backdoor studies on 3D point clouds. Different from the patch-based triggers in 2D images, they proposed customized triggers for point clouds, which include inserting point clusters around target objects [9], [20] and rotating target objects with a specific angle [9]. The above attacks only target the classification task on synthetic point clouds. The authors in [22] further extended the above idea to the LiADR object detection on outdoor point clouds. However, all of the existing methods adopt static triggers (*i.e.,* a single trigger pattern and a fixed trigger location) that could limit their effectiveness and flexibility in practical attack deployment. By contrast, we are the first to study the dynamic backdoor attack against LiDAR semantic segmentation task on outdoor point clouds.

IX. Conclusion

In this work, we make the first effort to reveal the backdoor vulnerability of LiDAR semantic segmentation in autonomous driving. To make the attack deployment more practical and flexible, we propose to construct a trigger pattern pool and a trigger location pool to realize the dynamic backdoor attack. Afterward, the attacker only needs to place a common object at an arbitrary location in the specified area to easily fool the backdoored model. We perform extensive experiments in both the digital world and the simulated physical world to demonstrate the effectiveness of our proposed method.

References

[1] lidar_simulation. https://github.com/jae251/lidar_simulation.

[2] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9297–9307, 2019.

[3] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z Morley Mao. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 2267–2281, 2019.

[4] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[5] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3075–3084, 2019.

[6] Xueluan Gong, Yanjiao Chen, Qian Wang, Huayang Huang, Lingshuo Meng, Chao Shen, and Qian Zhang. Defense-resistant backdoor attacks against deep neural networks in outsourced cloud environment. *IEEE Journal on Selected Areas in Communications*, 39(8):2617–2631, 2021.

[7] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[8] Xin Lai, Yukang Chen, Fanbin Lu, Jianhui Liu, and Jiaya Jia. Spherical transformer for lidar-based 3d recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17545–17555, 2023.

[9] Xinke Li, Zhirui Chen, Yue Zhao, Zekun Tong, Yabang Zhao, Andrew Lim, and Joey Tianyi Zhou. Pointba: Towards backdoor attacks in 3d point cloud. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16492–16501, 2021.

[10] Yiming Li, Yanjie Li, Yalei Lv, Yong Jiang, and Shu-Tao Xia. Hidden backdoor attack against semantic segmentation models. *arXiv preprint arXiv:2103.04038*, 2021.

[11] Yiming Li, Tongqing Zhai, Baoyuan Wu, Yong Jiang, Zhifeng Li, and Shutao Xia. Rethinking the trigger of backdoor attack. *arXiv preprint arXiv:2004.04692*, 2020.

[12] Youquan Liu, Yeqi Bai, Lingdong Kong, Runnan Chen, Yuenan Hou, Botian Shi, and Yikang Li. Pcseg: An open source point cloud segmentation codebase. https://github.com/PJLab-ADG/PCSeg, 2023.

[13] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Mārtiņš Možeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, et al. Lgsvl simulator: A high fidelity simulator for autonomous driving. In *2020 IEEE 23rd International conference on intelligent transportation systems (ITSC)*, pages 1–6. IEEE, 2020.

[14] Ahmed Salem, Rui Wen, Michael Backes, Shiqing Ma, and Yang Zhang. Dynamic backdoor attacks against machine learning models. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, pages 703–718. IEEE, 2022.

[15] Junjie Shen, Jun Yeon Won, Zeyuan Chen, and Qi Alfred Chen. Drift with devil: Security of {Multi-Sensor} fusion based localization in {High-Level} autonomous driving under {GPS} spoofing. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 931–948, 2020.

[16] Jiachen Sun, Yulong Cao, Qi Alfred Chen, and Z Morley Mao. Towards robust {LiDAR-based} perception in autonomous driving: General black-box adversarial sensor attack and countermeasures. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 877–894, 2020.

[17] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[18] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII*, pages 685–702. Springer, 2020.

[19] James Tu, Mengye Ren, Sivabalan Manivasagam, Ming Liang, Bin Yang, Richard Du, Frank Cheng, and Raquel Urtasun. Physically realizable adversarial examples for lidar object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13716–13725, 2020.

[20] Zhen Xiang, David J Miller, Siheng Chen, Xi Li, and George Kesidis. A backdoor attack against 3d point cloud classifiers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7597–7607, 2021.

[21] Jianyun Xu, Ruixiang Zhang, Jian Dou, Yushi Zhu, Jie Sun, and Shiliang Pu. Rpvnet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16024–16033, 2021.

[22] Yan Zhang, Yi Zhu, Zihao Liu, Chenglin Miao, Foad Hajiaghajani, Lu Su, and Chunming Qiao. Towards backdoor attacks against lidar object detection in autonomous driving. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*, pages 533–547, 2022.

[23] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9939–9948, 2021.

[24] Yi Zhu, Chenglin Miao, Foad Hajiaghajani, Mengdi Huai, Lu Su, and Chunming Qiao. Adversarial attacks against lidar semantic segmentation in autonomous driving. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, pages 329–342, 2021.

[25] Yi Zhu, Chenglin Miao, Tianhang Zheng, Foad Hajiaghajani, Lu Su, and Chunming Qiao. Can we use arbitrary objects to attack lidar perception in autonomous driving? In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 1945–1960, 2021.