IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 9 January 2023, accepted 25 January 2023, date of publication 31 January 2023, date of current version 3 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3241236

RESEARCH ARTICLE

Reliable Detection of Location Spoofing and Variation Attacks

CHIHO KIM[1], (Member, IEEE), SANG-YOON CHANG[2], (Member, IEEE), DONGEUN LEE[1], JONGHYUN KIM[3], KYUNGMIN PARK[3], AND JINOH KIM[1], (Senior Member, IEEE)

[1]Computer Science Department, Texas A&M University–Commerce, Commerce, TX 75035, USA
[2]Department of Computer Science, University of Colorado Colorado Springs, Colorado Springs, CO 80918, USA
[3]Cybersecurity Research Division, Electronics Telecommunications Research Institute (ETRI), Daejeon 34129, South Korea

Corresponding author: Jinoh Kim (Jinoh.Kim@tamuc.edu)

This work was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) through the Korea Government [Ministry of Science and ICT (MSIT)], Research on Foundational Technologies for 6G Autonomous Security-by-Design to Guarantee Constant Quality of Security, under Grant 2021-0-00796.

ABSTRACT Location spoofing is a critical attack in mobile communications. While several previous studies investigated the detection of location spoofing attacks, they are limited in their performance and lack the consideration of emerging attack variations. In this paper, we present a data-driven methodology for the reliable detection of location spoofing and its variations. To enhance the performance, we introduce and utilize a new set of features, which is differential in nature and enables the checking of the mobility constraints and inconsistency. Our comparison study with the previous research shows that the presented scheme using the new features significantly improves the accuracy and reliability of the detection against location spoofing attacks. To take the possibility of attack variations into account, we establish a set of scenarios manipulating coordinate data to create attack variants. Our experimental results confirm the feasibility and effectiveness of the new features for identifying diverse types of spoofing attacks and their variations, greatly improving the detection performance by up to 99.1% accuracy. Additionally, we present a profiling-based detection approach (building the detector referring only to legitimate coordinate data), to further extend resilience to previously unseen attacks as a means to zero-day detection. The evaluation result shows the potential of the profiling-based detector with comparable or even better performance than the supervised learning methods (requiring both legitimate and falsified data to construct the detector).

INDEX TERMS Attack variation, location spoofing, mobile communications, position falsification, profiling-based detection, zero-day detection.

I. INTRODUCTION

In mobile communication, the coordinate of mobile agents is the crucial information for various applications, such as performance prediction [1], resource allocation and offloading decisions [2], and mobile agent deployment and routing [3], to list a few. For instance, we may want to predict application throughput at the current position of a mobile device [4]. Another example would be that the unmanned aerial vehicle (UAV) can be deployed to a suitable place to provide the required quality of service [5]. The location information is also crucial for vehicular ad-hoc network (VANET) since it is directly connected to safety functions, such as collision avoidance and improper lane change detection [6], [7].

Given the importance of the coordinate information of mobile agents, protecting the integrity of such information is extremely vital. Unfortunately, however, there can be a chance of the exchange of incorrect coordinates, which leads to unwanted consequences owing to several reasons. For example, GPS hardware may malfunction. Even worse, position information can be manipulated intentionally. Location spoofing (or position falsification) refers to faking location so that an adversary masquerades as being at a different location [8], [9], [10], [11], [12]. Such an attack can also be used for launching other subsequent attacks. Since location spoofing poses serious threats to security, the detection of

The associate editor coordinating the review of this manuscript and approving it for publication was Li He.

falsified coordinates should be considered crucial to promote greater security for location-based mechanisms in mobile networks.

This paper explores the challenges of location spoofing attacks with a well-known mobile dataset providing coordinate information, Vehicular Reference Misbehavior (VeReMi) dataset [13], [14], [15]. We chose VeReMi since it provides location spoofing attack instances together with genuine samples with unmanipulated coordinates. Several previous studies [16], [17], [18], [19], [20] employed the VeReMi dataset to measure detection performance against spoofing attacks, and it was reported that conventional machine learning (ML) methods are able to yield highly accurate detection rates, e.g., >95% with a random forest classifier against five attack types [17]. However, our preliminary study using the same methodology and replicating the previous research reveals somewhat lower performance than what has been reported in previous studies, signaling the need for a thorough investigation into the problem of location spoofing attacks. More critically, previous studies concentrate on predefined attack vectors (e.g., five static types in [14]), which confines their efficacy. Our study shows that existing detection schemes are susceptible to attack variations even with minor modifications of coordinates, degrading detection performance considerably. Indeed, this is critical given the evolution of attacks with cleverer schemes to bypass detection functions.

Motivated by those observations and limitations, this study investigates location spoofing attacks in mobile communication settings and develops a data-driven methodology for reliable detection of different types of attack trials, including their potential variations. In our exploration, we place our focus on *detection performance*, *resilience to attack variations*, and *light-weight detection* without the need for heavy computational power. For detection performance, we examine the effectiveness of current feature sets defined in existing studies and present newly formulated features to enhance the performance. We also shed light on the impact of potential attack variations and evaluate the reliability of detection functions against such variants. Last but not least, the detection function is often required for concluding the decision in a timely manner, which is one of the principles placed on our design of the detection methodology.

The following is a summary of the key contributions of this study:

- We present a feature set that is differential in nature and enables the checking of the mobility constraints and inconsistency. Our comparison study reveals that the use of existing features results in poor performance, particularly in several attack scenarios. We show that the new feature set is greatly beneficial, with which detection rates consistently outperform those with existing features across different scenarios. From the experiments conducted on a fair setting (including the same training and testing sets), the new feature set yields a detection performance of up to 99.1%, significantly outperforming the existing feature set with 94.5% at best.
- We establish a set of scenarios to consider attack variations when manipulating coordinate data. Spoofing attacks may not be static as defined in the original dataset; rather, future attacks would be more intelligent when fabricating location information with dynamic choices of coordinate offsets. We share our evaluation results showing the capability of our methodology with the substantially enhanced resilience to attack variations.
- We further consider resilience to attack variance with a profiling-based detection approach as a means to zero-day detection of previously unseen patterns of attacks. We implement a profiling-based detector on top of the autoencoder architecture, which identifies attack samples by characterizing genuine benign samples. Our experimental results show that the profiling-based detector performs greatly with the proposed feature set, yielding comparable or even better performance than the supervised learning methods.

The organization of this paper is as follows. In Section II, we introduce the background of this study with a description of the problem tackled and the dataset employed for developing the detection scheme with the evaluation methodology. We then present a new feature set defined for improving detection performance in Section III, and a set of scenarios for creating attack variations will be introduced in Section IV. To consider the zero-day detection capability, Section V presents our profiling-based detection approach with its implementation using the variational autoencoder structure. We provide a brief of the closely related studies in Section VI and conclude our presentation in Section VII with a summary of this study and future directions.

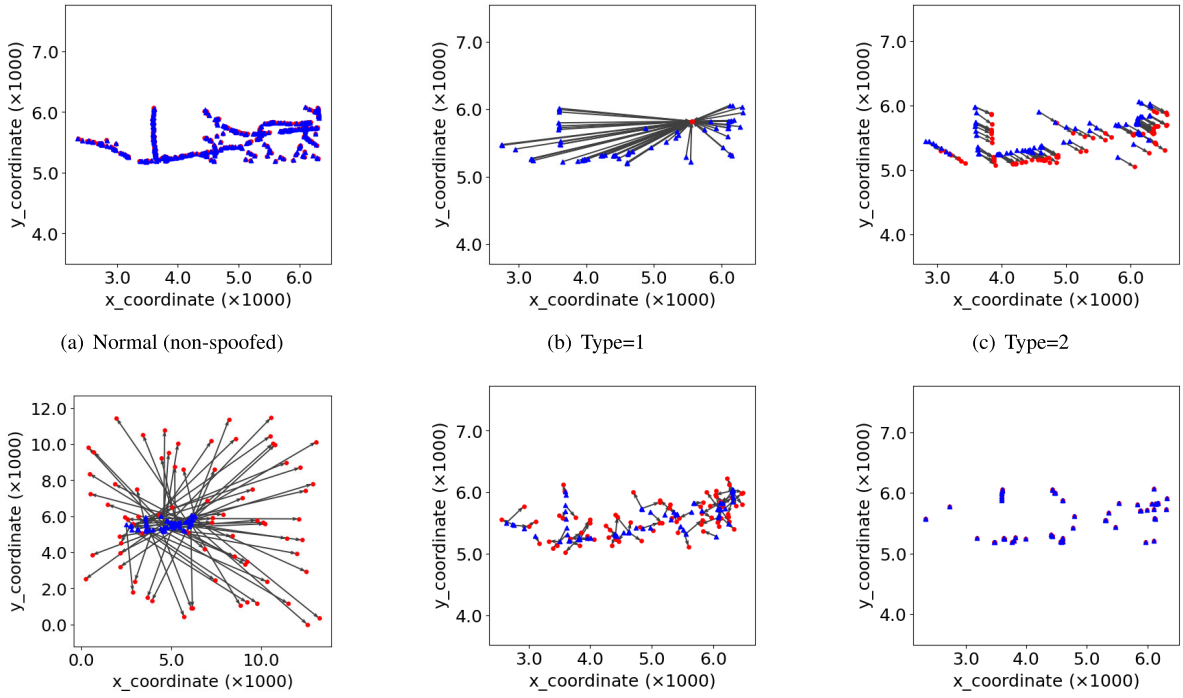
II. BACKGROUND

This section provides the overview and background of this study. We first describe the problem with the 2-sequence approach tackled in this study for light-weight detection of location spoofing attacks. We then provide the overview of the dataset employed for the development and evaluation of detection methods with a brief description of ML algorithms considered in this study. Finally, we introduce the experimental setting and metrics for performance comparison.

A. PROBLEM DESCRIPTION

This study adopts a simple definition of location spoofing: a true location of a mobile agent is not equal to a location advertised by that agent. The equality of the location information is tested with an acceptable margin (ϵ) to account for precision errors.

Formally, we describe our problem tackled in this study with notations summarized in Table 1. To detect an incorrectly advertised location (accidentally or intentionally), we assume the availability of location-related information for mobile agents, which contains the coordinate ($C = (c_x, c_y, c_z)$), velocity ($V = (v_x, v_y, v_z)$), and integer time



(a) Normal (non-spoofed) (b) Type=1 (c) Type=2

(d) Type=4 (e) Type=8 (f) Type=16

FIGURE 1. Location spoofing examples by attack types: The start point is represented in blue, the endpoint is in red, and the arrow shows the direction from the start point to the endpoint.

TABLE 1. Notations.

Notation	Description
C	3D coordinate (c_x, c_y, c_z)
V	Velocity in 3D space (v_x, v_y, v_z)
$\ominus$	Element-wise subtraction operator
D	Difference between two coordinates (d_x, d_y, d_z)
C^t	Coordinate of the mobile agent at time t
V^t	Velocity of the mobile agent at time t
D^t	Difference between C^{t-1} and C^t
d^t	Euclidean distance between C^{t-1} and C^t
S^t	State of the mobile agent at time t, $S^t = \langle C^t, V^t \rangle$
Δ^t	Time difference between two time steps of t and $t-1$
κ^t	Movement plausibility check (MPC) constant (derived from S^{t-1} and S^t)

TABLE 2. Attack types defined in the dataset.

Type	Attack name	Parameters
1	Constant	$x = 5560, y = 5820$
2	Constant_ offset	$\Delta x = +250, \Delta y = -150$
4	Random	random coordinate
8	Random_ offset	$\Delta x, \Delta y \leftarrow [-300, +300]$
16	Eventual_stop	Stop probability increases by 0.025 at each position update

step t, where (x, y, z) denotes the three-dimensional space.[1] To compare two different coordinates, C_i and C_j $(i \neq j)$, we define $D_{i,j} = (d_x, d_y, d_z) = C_j \ominus C_i$, where $\ominus$ is an element-wise subtraction operator. For each element in $D_{i,j}$, a positive value indicates the forward direction in that axis, and the following holds: $D_{i,j} = -D_{j,i}$. We assume that $D_{i,j} = 0$ if all element values are lower than the acceptable margin (i.e., $d_x < \epsilon$, $d_y < \epsilon$, and $d_z < \epsilon$).

Similarly, we define the location-wise state of a mobile agent to keep track of it over time. The state of an agent is defined as $S^t = \langle C^t, V^t \rangle$, where C^t and V^t are the coordinate and the velocity at time t. The position change at t (i.e., the difference between two consecutive time steps from $t-1$ to t) is defined as $D^t = C^{t-1} \ominus C^t$, and $D^t = 0$ indicates no movement (within the acceptable margin) in that time interval.

Based on this, we focus on identifying whether the currently advertised location C^t is true (*Normal*) or not (*Spoofed*), by referring to two consecutive states (S^{t-1} and S^t). Hence, it is considered a "2-sequence" approach that detects location spoofing by referencing the location information collected from two consecutive advertisements. We note that it is also possible to consider an n-sequence approach ($n > 2$) to expect improved detection rates, as introduced in [16]. However, it takes more time to collect the required number of advertisements with increasing analysis costs if $n > 2$, while our scheme performs well with high accuracy with $n = 2$.

This study considers the binary classification that determines whether an updated location is correct (genuine) or not. There can be different types of spoofing attacks, as shown in Table 2. While it may be interesting to identify the type of individual spoofing attacks, this study focuses on the determination of the correctness of the advertised location

[1] Although we assume a three-dimensional space for generality, including aerial agents, it can be simply reduced to a two-dimensional space by setting c_z and v_z to zeros.

information. The rationale of this is a specific type of location spoofing attack may not trigger a prescribed response, unlike intrusion detection functions often expected to enforce a relevant security policy (e.g., filtering, logging, etc.) against the detected event type.

B. DATASET DESCRIPTION

To develop and evaluate detection functions against location spoofing attacks, we employ the VeReMi dataset, which includes a collection of data instances with original and spoofed coordinate information [14], [15]. In VANET, each vehicle broadcasts its information in the form of Basic Safety Messages (BSMs), which occur at regular intervals (e.g., 10 times per second) [21]. One of the critical concerns in VANET is that malicious agents may inject falsified information through BSMs, which can adversely affect other agents. VeReMi provides a log of BSMs, each of which contains the coordinate, velocity, and timestamp information. The dataset includes 225 simulations assuming different traffic scenarios with data instances for both genuine and falsified positions.

The VeReMi dataset defines five spoofing attack types (1, 2, 4, 8, and 16), as summarized in Table 2. In our context, the constant attack (Type=1) sets C^t to the predefined coordinate, while the constant_offset attack (Type=2) changes the coordinate with the fixed Δx and Δy values. The randomization-based attacks (Type=4 and Type=8) are similar to the constant-based attacks (Type=1 and Type=2), but they choose the parameter values randomly. Lastly, the eventual_stop attack (Type=16) is to mimic as if the mobile agent had stopped based on increasing stop probability.

To take a closer look at the different attack types, Fig. 1 visualizes how these attacks change the original coordinates to false positions. In the figure, the arrow shows the direction from a start point (in blue) to an endpoint (in red). For example, we can see that all the genuine points (in blue) are modified to the single location (in red) in case of Type=1 (constant attack), while the genuine and modified points are the same for Type=16 (eventual_stop attack). Additionally, Type=4 (random attack) shows a wider coordinate space (than the original space) with randomized modified points.

C. MACHINE LEARNING METHODS

In this study, we evaluate detection performance with a set of widely-applied classification algorithms. These ML methods were also considered in previous studies [16], [17], [18], [19], [20] (but we exclude under-performing algorithms, such as decision trees, naïve Bayes, and linear regression). We additionally consider a neural network model, which is capable of dealing with non-linearity. While the details of the ML algorithms can be found from [22], [23], the following is a brief description of the algorithms:

- *k-Nearest Neighbors* (KNN): An instance-based learning algorithm characterized by memorizing training instances. KNN performs the classification by finding the k nearest neighbors of the sample using a distance function (e.g., Euclidean distance). The prediction is made based on majority voting among the nearest neighbors.
- *Support Vector Machine* (SVM): An extension of the perceptron algorithm. Unlike the basic perceptron algorithm minimizing misclassification errors, SVM maximizes the margin between the decision boundary (hyperplane) and instances closest to the hyperplane (support vectors).
- *Random Forest* (RF): An ensemble of decision trees. RF divides the input space into multiple subsets, which are fed into individual decision trees running in parallel. The final decision is then made by combining the prediction outcome of the trees (e.g., taking an average or using a majority vote).
- *Extreme Gradient Boosting* (XGB): An ensemble method based on a distributed gradient-boosted decision tree. Gradient boosting assumes a gradient descent algorithm over an objective function to combine individual decision trees sequentially. Each tree is built to reduce the residuals of the previous tree along the sequential process. XGB is a scalable implementation of gradient boosting.
- *Multi-layer Perceptron* (MLP): A multi-layer feed-forward neural network consisting of an input layer, one or more hidden layer(s), and an output layer. MLP is trained to perform input-output mapping, often with a non-linear (activation) function.

While we can simply take default settings for most ML algorithms above, a neural network should be organized and configured to implement MLP. For this purpose, we created an MLP structure with three hidden layers with the activation function of `ReLU` (in the first two hidden layers) and `Sigmoid` in the last hidden layer, which is connected to the `Softmax` function. The loss is measured with the cross-entropy function for binary classification.

D. EVALUATION SETTING AND METRICS

For evaluation, we transform the original dataset into a 2-sequence collection. Any two adjacent messages advertised by the same sender are combined and form an instance with S^{t-1} (the former message) and S^t (the latter message). We then organize two disjoint sets for training and testing from the 2-sequence collection. Each set contains an equal number of instances for Normal and Spoofed. There are 8,000 instances for training and 2,000 for testing. For the Spoofed instances, each attack type has an equal number of instances.

In this study, we focus on binary classification to determine whether an advertised location is legitimate or not. To measure detection performance, we mainly use the metric of accuracy. It is known that the accuracy may lead to a biased conclusion if there is a class imbalance (i.e., a significant difference between the majority and minority classes in quantity). In case of the unbalanced setting, the metric of F1 score based on the harmonic mean would be more

TABLE 3. Performance metrics.

Metric	Definition
True Positive (TP)	Actually falsified and predicted correctly
True Negative (TN)	Actually genuine and predicted correctly
False Negative (FN)	Actually falsified but predicted incorrectly
False Positive (FP)	Actually genuine but predicted incorrectly
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$
False Positive Rate (FPR)	$\frac{FP}{TN+FP}$
False Negative Rate (FNR)	$\frac{FN}{TP+FN}$
Precision	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$
F1 score	$2 \times \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$

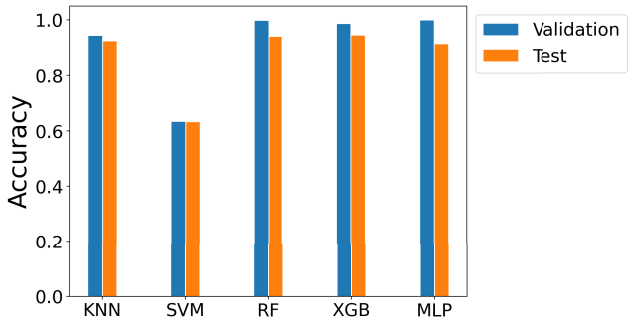


FIGURE 2. Validation and testing performance (Basic): All the detectors other than SVM perform well with greater than 94.4% of accuracy at validation, which drops down to 91.4% at testing. SVM performs poorly with only 63% of accuracy. The highest accuracy measured at testing is 94.5% by XGB.

reliable for measuring the performance. When the testing set is balanced with an equal number of Normal and Spoofed instances, it is safe to rely on the accuracy metric to report the performance of detectors. In other cases, we will use the F1 score. Table 3 provides a description of the standard metrics for classification.

We also evaluate the detection models with attack instances only and see performance for individual attack types. In that case, we report the detection rate, which is simply a fraction of detected instances out of entire attack samples.

III. FEATURE ENGINEERING

Feature engineering is one of the core parts in data preparation, developing features from raw data [24]. In this section, we first examine the effectiveness of the basic 2-sequence feature set consisting of S^{t-1} and S^t (named 'Basic'). We then present our proposed feature set (named 'Ext'), defining a set of new features with the intuition behind them. The performance of Basic and Ext will be compared and reported.

A. BASIC FEATURE SET

As mentioned, we consider two consecutive advertisements from the mobile agent for the detection task. Thus the two back-to-back BSMs form a single data instance in the context of the VeReMi dataset. As a result, the Basic feature set contains $\langle S^{t-1}, S^t \rangle$. Note that the same type of data transformation has been considered in previous studies [17], [18], [25].

TABLE 4. Detection rate breakdown by attack types (feature set=Basic).

Classifier	Type=1	Type=2	Type=4	Type=8	Type=16
KNN	0.985	0.990	0.995	0.735	0.710
SVM	0.510	0.000	0.825	0.000	0.000
RF	0.990	0.970	0.995	0.840	0.735
XGB	0.990	0.990	0.995	0.885	0.700
MLP	0.985	1.000	0.995	1.000	1.000

We first look at the effectiveness of the Basic feature set empirically. Fig. 2 shows the validation and testing accuracy performed with Basic. Here, the validation performance refers to classification accuracy measured with the training set. While their overall performance is good, we found that the testing accuracy is somewhat lower than the validation accuracy in our experimental setting. For instance, RF and XGB show quite degraded performance from 99% (at validation) to around 94% (at testing). KNN follows with 92.4% testing accuracy, while SVM works quite poorly with around 63% accuracy for both validation and testing. Finally, MLP performs great with 100% at validation, but it goes down to 91.4% at actual testing.

To further analyze, we take a closer look at the detection performance against individual attack types. Table 4 shows the detection performance breakdown by attack types obtained with Basic. From the table, we can see that some attack types are relatively well detected, while some other types show degraded detection rates. When utilizing KNN, RF, and XGB, for example, the classifiers produce at least 97% of detection performance for Types 1, 2, and 4, whereas it drops to 88.5% and less for Type 8 and 16. SVM performs unsatisfactorily with extremely low detection rates. Interestingly, MLP shows consistent performance across the different attack types, with 98.5% of detection rates or higher. By combining the observation made in Fig. 2, MLP detects attack instances very well at the expense of a non-negligible false positive rate, limiting its overall performance to 91.4% accuracy.

B. PROPOSED FEATURE SET

While the Basic feature set is compact, it is limited with 94.5% accuracy at max (by XGB) due to the unacceptable performance to detect certain types of spoofing attacks. A potential weakness of Basic is its dependency on real coordinates. As a result, even a slight difference in the coordinate space may diminish the capability of detection models.

In this study, we define a new feature set to better capture the characteristics of normal vs. spoofed data instances. Here, our objective is to define features that can help improve detection performance with a manageable cost to compute new features so as to support accurate, timely detection of

spoofing attacks. To meet this objective, we create a set of new features by transforming the Basic features, which enables us to capture the checking of the mobility constraints and inconsistency, as follows:

- D^t: Difference between two consecutive coordinates (i.e., $C^{t-1} \ominus C^t$);
- d^t: Euclidean distance between two consecutive coordinates of C^{t-1} and C^t;
- Δ^t: Time gap between two consecutive advertisements (i.e., the difference between two time steps of t and t-1)
- κ^t: Movement plausibility check (MPC) constant (derived from S^{t-1} and S^t).

The intuition behind the definition of D^t and d^t is to relax the limitation of the reliance on real coordinates by measuring the movement along the axes (D^t) and the moving distance over the Euclidean space (d^t). The feature of time gap (Δ^t) may be crucial information if the advertisement can be missing or irregular for some reason. The concept of movement plausibility check (MPC) constant is borrowed from the previous study in [16], defined to detect the discordance between the location and velocity information. In detail, the MPC feature records whether the location remains the same despite a non-zero velocity value (i.e., actually moved). We define MPC at t (κ^t), as follows:

$$\kappa^t = K \times ((V^{t-1} \neq 0) \wedge (C^{t-1} = C^t))$$

Here, K is a pre-selected constant (e.g., K=1000 in [16]).

We define the Ext set to include these new features beyond the Basic set. The new features are differential in nature. Our intuitions for introducing them to the detection problem are that the law of physics bounds the spatial state changes (e.g., cannot teleport) and that there cannot be inconsistencies due to dataset tampering. Introducing these new features enables the checking of such intuitions. We later show that the differential features in Ext significantly improve the detection performance and robustness across the attack variations.

There would be some other features that may help improve detection rates. In [16], for example, two additional features of Minimum Distance to Trajectories (MDT) and Minimum Translation Distance to Trajectories (MTDT) are defined in addition to MPC. The feature of MDT searches the closest data instance to the current one by measuring the Euclidean distance over the trajectory. The MTDT feature is similar to MDT, but the difference is that MTDT relies on a translation vector to search the closest data instance to the current sample in question. These features are computationally too expensive, requiring aggregations and pair-wise computations with the entire legitimate samples in the training set. Since feature engineering complexity is a crucial element for supporting real-time operations, particularly under the provision of restrictive resources, we do not consider these features imposing too expensive engineering costs. Note that the engineering cost for our defined features is $O(1)$, since they can be calculated directly from the existing features in that instance (without referring to any other instances).

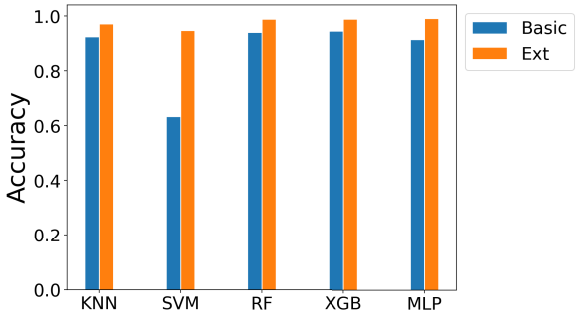


FIGURE 3. Performance comparison between Basic and Ext: Using the Ext feature set greatly improves the detection performance from 4.3% (XGB) to 31.4% (SVM) compared to Basic. All the detectors yield at least 94.7% accuracy with Ext, and MLP shows the best, producing 99.1% accuracy.

TABLE 5. Detection rates breakdown by attack types (feature set=Ext).

Classifier	Type=1	Type=2	Type=4	Type=8	Type=16
KNN	0.985	0.990	0.995	0.835	0.935
SVM	0.945	0.990	0.995	0.620	0.920
RF	0.985	0.990	0.995	0.950	0.970
XGB	0.985	0.990	0.995	0.960	0.970
MLP	0.995	1.000	0.920	1.000	1.000

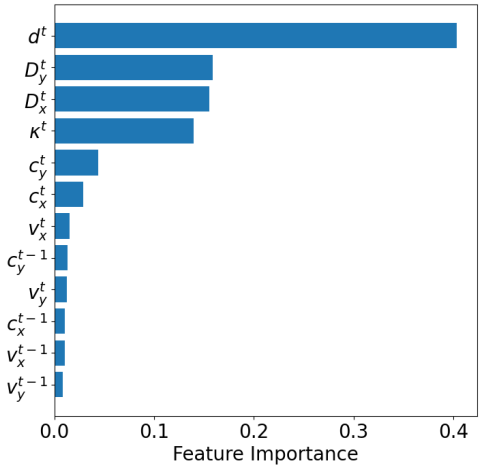


FIGURE 4. Significance of features (compiled by RF): The newly defined features in Ext play a crucial role in the detection, placed on higher ranks than the existing features.

We now examine the performance with the Ext feature set. Fig. 3 compares the classification accuracy measured with the Basic and Ext feature sets. The figure shows that the Ext feature set greatly helps improve the detection performance. Even the SVM classifier yields 94.7% with Ext, while it was 63.3% when using the Basic set. Other than SVM, all the detectors yield at least 94.7% accuracy with Ext, and MLP shows the best producing, 99.1% accuracy.

Table 5 presents the performance breakdown by attack types as a comparison to the result in Table 4. Overall, the detection performance has significantly been enhanced with Ext. In particular, relying on the new feature set is greatly helpful for detecting Type 8 and 16 attacks. The detectors based on the XGB and RF models perform consistently across the attack types with 95% detection rates at the minimum.

TABLE 6. Performance details for Basic and Ext.

Set	Classifier	TPR	TNR	FPR	FNR	Accuracy	Precision	Recall	F1 score
BASIC	KNN	0.883	0.965	0.035	0.117	0.924	0.961	0.883	0.920
	SVM	0.267	1.000	0.000	0.733	0.633	1.000	0.267	0.421
	RF	0.906	0.974	0.026	0.094	0.940	0.972	0.906	0.937
	XGB	0.912	0.978	0.022	0.088	0.945	0.976	0.912	0.943
	MLP	0.996	0.832	0.168	0.004	0.914	0.855	0.996	0.920
EXT	KNN	0.948	0.994	0.006	0.052	0.971	0.993	0.948	0.970
	SVM	0.894	1.000	0.000	0.106	0.947	1.000	0.894	0.944
	RF	0.978	0.998	0.002	0.022	0.988	0.998	0.978	0.987
	XGB	0.980	0.997	0.003	0.020	0.988	0.996	0.980	0.988
	MLP	0.983	1.000	0.000	0.017	0.991	1.000	0.983	0.991

MLP shows a slightly lower detection rate for Type 4 attacks with Ext than with Basic, while it deals with the other types of attacks very well. Nonetheless, the overall performance of MLP (in Fig. 3) shows that it performs better with Ext than Basic.

To see the significance of the features defined in the Ext set, Fig. 4 provides the ranks measured by the RF algorithm. Interestingly, the Euclidean distance (d^t) plays a vital role in detection. The difference measures in the x-axis (D_x^t) and the y-axis (D_y^t) between two sequences follow next. In the VeReMi dataset, D_z^t has no effect since the movement takes place in a two-dimensional space; however, this feature would be crucial in other domains assuming the movement in a three-dimensional space. We also see that MPC (κ^t) is ranked higher than the features in Basic. *The feature importance indicates that the new features defined in the* Ext *set contribute to the significant improvement of performance in the detection of spoofing attacks.* Note that the other features have negligible significance values (less than 0.1) and are omitted.

Summary: Table 6 summarizes the performance of the detection models in detail. As can be seen, utilizing the Ext set shows much lower false rates (FPR and FNR) than Basic. The only exception is that MLP shows a slightly higher FNR with Ext, implying its tendency to classify instances more to Normal, considering its zero FPR with Ext. Lastly, the metrics of accuracy and F1 score produce similar results since the test set is well balanced with an equal number of Normal vs. Spoofed instances.

IV. RESILIENCE TO ATTACK VARIATIONS

While previous studies referred to the five spoofing attack types described in Table 2, it is not hard to assume the emergence of attack variants, for example, modifying values of the parameters defined for each attack. In this section, we establish a set of scenarios based on parameter modification to consider attack variations when manipulating coordinate data. We then evaluate the impact of the modification schemes on detection performance.

A. PARAMETER MODIFICATION

To measure the resilience to attack variations, we establish four scenarios that modify the parameter values defined in Table 2, as follows:

- Type 1 (*Constant*): The original dataset moves the position to the predefined single coordinate ($x = 5560$, $y = 5820$) for any attack instance under this type. The modified encoding applies a range of variations of $x = (5560 + \alpha_x)$ and $y = (5820 + \alpha_y)$. We vary α_x and α_y within the range of $[-300, +300]$, with a step size $\delta = 50$.
- Type 2 (*Constant_offset*): The basic offset values are set to $\Delta x = +250$ and $\Delta y = -150$. In our variation scenario, we vary the offset values from -300 to $+300$ for both Δx and Δy, with a step size $\delta = 50$. Hence, the offset value ranges become $\Delta x \leftarrow [-300, +300]$ and $\Delta y \leftarrow [-300, +300]$.
- Type 4 (*Random*): In the original dataset, the random coordinate space ($x \leftarrow [0, 14000]$ and $y \leftarrow [0, 12000]$) is too larger than the actual coordinate range ($x \leftarrow [2000, 7000]$ and $y \leftarrow [5000, 6500]$). We retain the same coordinate space for generating random locations. From Fig. 1(a), however, we exclude the space between [5000, 6500] in the y-axis, so as to remove the possibility that the random location overlaps with any of the genuine coordinates.
- Type 8 (*Random_ offset*): Basically, the offset values are set to $\Delta x, \Delta y \leftarrow [-300, +300]$, within which random values are chosen for modifying the original x and y coordinates. We extend the offset ranges $\Delta x, \Delta y \leftarrow [-\beta, \beta]$, by varying the parameter β from 100 to 500 with a step size $\delta = 100$. Hence, the extended offset configuration varies from $\Delta x, \Delta y \leftarrow [-100, +100]$ to $\Delta x, \Delta y \leftarrow [-500, +500]$.

Note that Type 16 (*Eventual_stop*) is not considered for creating attack variations in our 2-sequence approach.

B. PERFORMANCE AGAINST ATTACK VARIATIONS

We evaluate the impact of the attack variations on detection performance with the Basic and Ext feature sets.

At first, Fig. 5 compares the impact of the Type 1 attack variations configured by varying α_x (in the x-axis) and α_y (in the y-axis, for each detector model. In the figure, a lighter color implies a better performance, while a darker color is for lower performance. Here, $(\alpha_x, \alpha_y) = (0, 0)$ indicates the predefined Type 1 attack defined in the original dataset, and the default Type 1 point shows almost the perfect detection rate across the different detectors, implying the detection

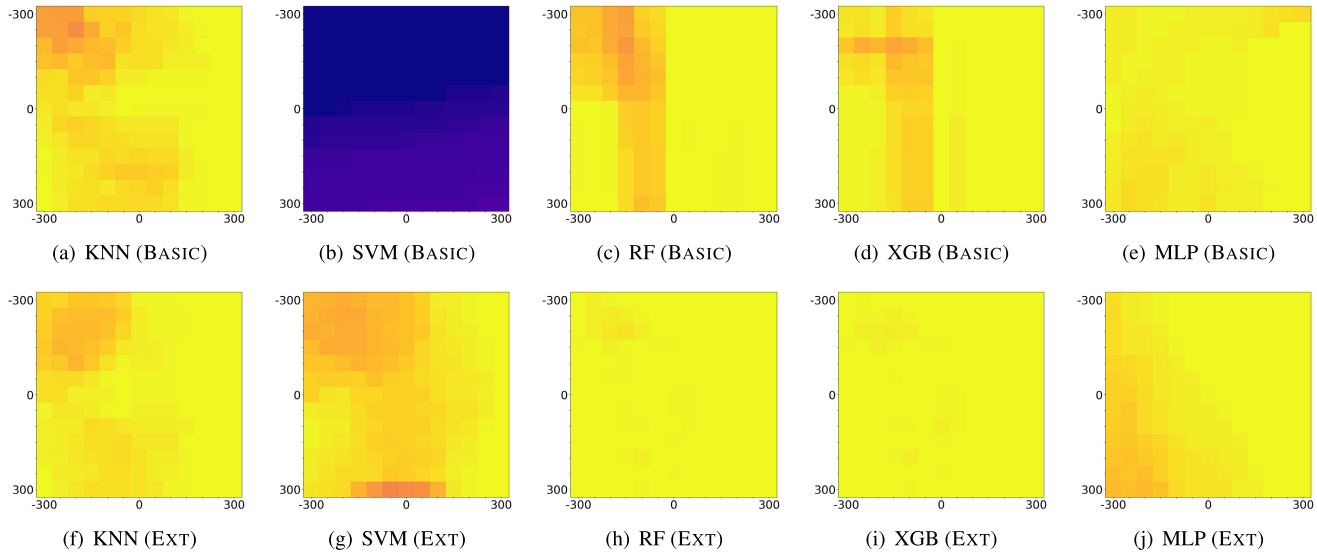


FIGURE 5. Performance against attack variations (Type=1): A lighter (yellow) color represents a higher detection rate, while a darker color does a lower rate. Overall, Ext outperforms Basic, with the aggregated average detection rate (across the detectors) of 98.4% (Ext) vs. 88.1% (Basic). When counting RF, XGB, and MLP only, the average rate is 98.4% (Basic) and 99.3% (Ext).

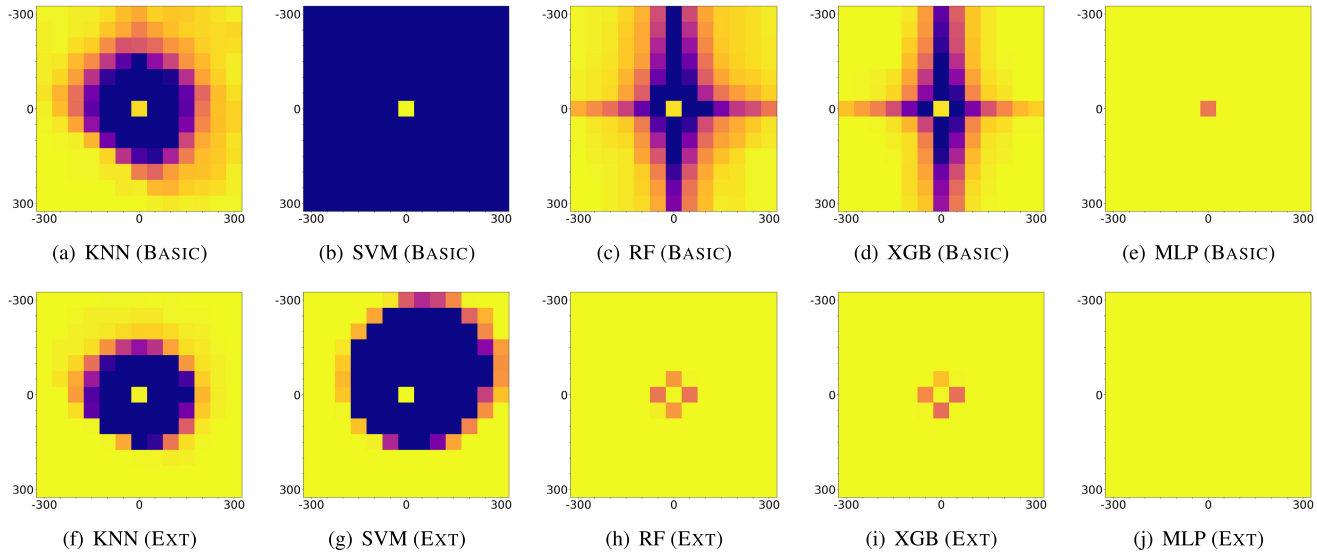


FIGURE 6. Performance against attack variations (Type=2): Type 2 attack variants have greater impacts on KNN and SVM (than Type 1 variations) when the offset is close to zero (i.e., closer to the genuine point). With Ext, MLP perfectly detects the variations, while RF and XGB are slightly degraded only if either α_x or α_y is immediately adjacent to zero. Ext performs better with the aggregated average detection rate of 89.2% than Basic with 72.5% across the detection models. If we count RF, XGB, and MLP only, the average rate observed is 91.7% (Basic) and 99.7% (Ext).

models are quite well trained. The overall result shows that using Ext produces much lighter colors (and hence, better detection rates) across the parameter value ranges for α_x and α_y.

For details, Table 7 shows the percentile of detection rates measured against Type 1 variants. In the table, we use "-%" to indicate percentile (e.g., x-% = x percentile). Other than SVM, using Basic performs well even in 10-%, showing 93.5% at the lowest. Using Ext performs much better, and even SVM produces a 91.5% detection rate at 10-%. The aggregated average detection rate across the detectors is 88.1% (Basic) and 98.4% (Ext). When counting RF, XGB, and MLP only, the average rate observed is 98.4% (Basic) and 99.3% (Ext).

We next examine the impact of Type 2 attack variations, and Fig. 6 shows the experimental result. The parameter setting of $(\alpha_x, \alpha_y) = (0, 0)$ indicates no use of offsets (since $\Delta x = \Delta y = 0$). As seen from the figure, Type 2 attack variants have greater impacts on KNN and SVM (than Type 1 variations in Fig. 5) when the offset is close to zero (i.e., closer to the genuine point). With Ext, MLP perfectly detects the variations, while RF and XGB are slightly degraded only if

TABLE 7. Percentile ("-%") of detection rates against attack variations (Type=1).

Set	Classifier	10-%	25-%	50-%	75-%	90-%	Mean
BASIC	KNN	0.940	0.965	0.980	0.995	1.000	0.973
	SVM	0.390	0.415	0.520	0.545	0.550	0.484
	RF	0.935	0.955	1.000	1.000	1.000	0.978
	XGB	0.950	0.970	0.995	1.000	1.000	0.984
	MLP	0.975	0.985	0.990	1.000	1.000	0.990
EXT	KNN	0.945	0.970	0.990	1.000	1.000	0.981
	SVM	0.915	0.945	0.965	0.985	1.000	0.960
	RF	0.995	1.000	1.000	1.000	1.000	0.998
	XGB	0.995	1.000	1.000	1.000	1.000	0.998
	MLP	0.955	0.975	0.995	1.000	1.000	0.984

TABLE 8. Percentile ("-%") of detection rates against attack variations (Type=2).

Set	Classifier	10-%	25-%	50-%	75-%	90-%	Mean
BASIC	KNN	0.343	0.770	0.940	0.985	0.996	0.817
	SVM	0.000	0.000	0.000	0.000	0.000	0.005
	RF	0.568	0.790	0.945	0.980	0.995	0.852
	XGB	0.638	0.880	0.980	1.000	1.000	0.900
	MLP	1.000	1.000	1.000	1.000	1.000	0.999
EXT	KNN	0.099	0.905	0.995	1.000	1.000	0.834
	SVM	0.000	0.000	0.945	1.000	1.000	0.634
	RF	1.000	1.000	1.000	1.000	1.000	0.996
	XGB	1.000	1.000	1.000	1.000	1.000	0.996
	MLP	1.000	1.000	1.000	1.000	1.000	1.000

TABLE 9. Performance against attack variations (Type=4).

Set	Classifier	Detection rate
BASIC	KNN	0.985
	SVM	0.305
	RF	0.995
	XGB	0.995
	MLP	1.000
EXT	KNN	1.000
	SVM	0.995
	RF	1.000
	XGB	1.000
	MLP	0.955

either α_x or α_y is immediately adjacent to zero. Table 8 shows the percentile of the detection rates against Type 2 variants. From the table, we can confirm that Ext performs better with the aggregated average detection rate of 89.2% than Basic with 72.5% across the detection models. If we count RF, XGB, and MLP only, the average rate is 91.7% (Basic) and 99.7% (Ext).

We move to Type 4 attack variations. The experimental result in Table 9 shows that Basic and Ext perform almost comparably, and the detection rates are pretty satisfactory. The only exception is the SVM detector showing an unacceptable performance (30.5% accuracy) with Basic, which yields 99.5% with Ext on the contrary.

Lastly, Table 10 shows the detection rates against Type 8 attack instances encoded with different β values. Here, $\beta = 300$ is the default setting applied in the original dataset. A smaller β value indicates a closer point to the original location, and hence, it may be more challenging to detect. We can see that RF, XGB, and MLP identify Type 8 variants very well with Ext even in case of $\beta = 100$, which is the most challenging parameter setting in Type 8. Using Basic works well with the MLP detector, while RF and XGB show clearly lower detection rates than Ext. We can see that KNN and SVM perform much worse than the other detection models, although Ext much enhances the performance compared to Basic.

TABLE 10. Detection rates against attack variations (Type=8).

Set	Classifier	β=100	β=200	β=300	β=400	β=500
BASIC	KNN	0.160	0.565	0.735	0.870	0.915
	SVM	0.000	0.000	0.000	0.000	0.000
	RF	0.470	0.720	0.840	0.885	0.910
	XGB	0.570	0.770	0.885	0.900	0.920
	MLP	1.000	1.000	1.000	1.000	1.000
EXT	KNN	0.150	0.565	0.835	0.940	0.975
	SVM	0.000	0.250	0.620	0.840	0.900
	RF	0.915	0.985	0.950	0.990	0.990
	XGB	0.935	0.985	0.960	0.990	0.990
	MLP	1.000	1.000	1.000	1.000	1.000

Summary: Through the evaluation performed with attack variations, we observe that using Ext consistently outperforms Basic, indicating greater resilience to attack variants emerging in the future. For Type 1 variations, the two feature sets yield the aggregated average detection rate (across the entire detectors) of 88.1% (Basic) vs. 98.4% (Ext) over the different classifiers on the diverse parameter setting. Similarly, Ext performs better with the aggregated average detection rate of 89.2% than Basic with 72.5% across the detection models. Using Ext keeps working better with 99% mean detection performance for Type 4 variations, while Basic degrades to 85.6% on the same setting. For Type 8, we observe that RF, XGB, and MLP identify its variants with the least performance of 91.5%, while Basic performs ineffectively, dropping the detection rate down to 47.0%.

V. ZERO-DAY DETECTION METHODOLOGY

We thus far concentrated on *supervised* learning for creating a detection model with an assumption of the availability of both Normal and Spoofed instances. Alternatively, a profiling-based approach can be considered for building up a detection model only using the Normal instances. The constructed profile can then be referenced to identify attack instances that are not conforming to the learned representation. The term *zero-day* detection refers to the identification of previously unseen attack instances different from the existing (known) representation [26], [27], [28], [29]. In this section, we present our approach to zero-day detection, given the emergence of new types of spoofing attacks with smarter features to fool the detector (e.g., using a generative model like generative adversarial networks [30]).

The concept of *semi-supervised* learning is the creation of the detection model only from the single data class [31], [32]. That is, we can construct a profile that captures the

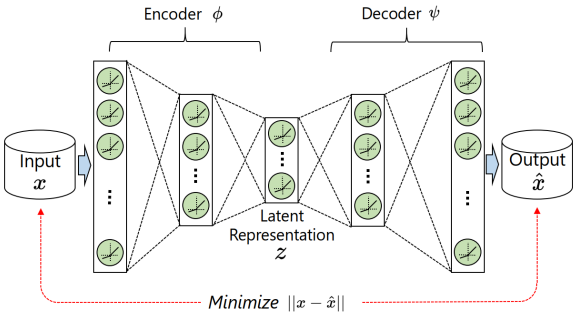


FIGURE 7. Autoencoder architecture: The encoder maps the input data into a low-dimensional space ($\phi : \mathbb{R}^n \rightarrow \mathbb{R}^m$, $n > m$), and the decoder regenerates from the latent representation to output ($\psi : \mathbb{R}^m \rightarrow \mathbb{R}^n$). Typically, the autoencoder minimizes the reconstruction error ($\epsilon := \|\vec{x} - \hat{\vec{x}}\|$) through the learning process.

characteristics of the Normal instances by relying on the semi-supervised detection concept. The autoencoder architecture has been utilized for implementing semi-supervised learning [33], [34], [35], [36]. The basic idea is that the autoencoder model restores Normal instances relatively well compared to Spoofed instances since it focuses on restoring Normal instances only in the data-learning phase.

A. AUTOENCODER ARCHITECTURE

We introduce the typical autoencoder architecture with its main functionality. As illustrated in Fig. 7, the autoencoder is composed of an encoder ($\phi(\cdot)$) and a decoder ($\psi(\cdot)$) with internal hidden layers. The encoder projects the input data to a lower dimensional space ($\phi : \vec{x} \rightarrow \vec{z}$), and the decoder restores the lower dimensional representation to the output on the input dimension space ($\psi : \vec{z} \rightarrow \hat{\vec{x}}$). The difference between the input ($\vec{x}$) and the output ($\hat{\vec{x}}$) is known as reconstruction error, defined as: $\epsilon := \|\vec{x} - \hat{\vec{x}}\|$. The autoencoder is then trained with a given dataset in a way to minimize the reconstruction error.

The autoencoder can be utilized for the purpose of *profiling*, capturing the unique characteristics of a certain class [36], [37]. For example, an autoencoder model can be established using training instances belonging to a single class under consideration. The autoencoder model would then be clever enough to restore a sample within that class with a bounded ϵ (reconstruction error). On the contrary, it has little idea of restoring a sample from a different class, resulting in a relatively larger ϵ. Based on the measured ϵ, the decision function ($\pi(\cdot)$) tests if the sample belongs to the class in question, e.g., π : if $\epsilon \leq \tau$, then True; else False, where τ is a pre-determined threshold (often chosen based on the distribution of reconstruction errors collected in the training stage).

B. PROFILING-BASED DETECTOR

We design the profiling function using a variational autoencoder (VAE). The VAE maintains the probabilistic information, which is referred to in the generative process. Internally, the probabilistic information is collected from the encoded representation, and the output is regenerated based on sampling provided by the sampling layer placed between the encoder (ϕ) and decoder (ψ).

We configure the encoder with the same number of hidden layers used in MLP, while the decoder has the mirrored structure of the encoder (as illustrated in Fig. 7). The VAE is equipped with `LeakyReLU` for internal non-linearity and `tanh` for the last-mile activation. For profiling, the VAE is trained with genuine instances (Normal) only. Like the MLP model chosen in this study, the VAE model with the least loss has been selected for performance evaluation, which makes a decision for a test sample whether it is Normal or not, based on the measured reconstruction error (ϵ). To set up the threshold (τ) for actual determination, we simply take the percentile information from the distribution of reconstruction errors (gathered in the training time). The underlying idea here is that taking $\tau = x\%$ means that the expected (and tolerable) FPR is $(1 - \frac{x}{100})$, since τ comes from the 95-percentile out of the ϵ distribution. For example, if $x = 95\%$, the upper 5% of the instances with greater reconstruction errors in the training set should be identified as non-Normal, which results in FPR=5%. We examine the model with a set of threshold values: $\tau = \{95\%, 96\%, 97\%, 98\%, 99\%\}$.

C. PERFORMANCE EVALUATION

To evaluate the performance of the profiling-based detection using the autoencoding concept, we use exactly the same experimental setting. The only difference is that the training set excludes the Spoofed instances (unlike the supervised learners trained with both Normal and Spoofed instances).

Table 11 summarizes the performance of profiling-based detection measured with Basic and Ext. The result shows that Basic performs unsatisfactorily with lower than 70% accuracy with any threshold setting. In contrast, Ext boosts the performance dramatically, yielding over 98% accuracy across different threshold values. Interestingly, the profiling-based detector equipped with Ext is comparable to or even better than the supervised learners in Table 6.

We next evaluate the resilience to attack variations. The same set of attack variations presented in Section IV is used for evaluating the resilience of the profiling-based detector. In this experiment, we simply set $\tau = 99\%$, since the performance is not significantly different across the threshold values (as seen from Table 11). Table 12 shows the detection rates of the profiling-based detector implemented with VAE. Here, we report detection rates for Type 4 and average detection rates for Type 1, 2, and 8 (aggregated over their parameter value range). We can see that our VAE-based detector performs very well with Ext, producing at least a 93% detection rate for attack variations. In particular, it shows better performance than the supervised learning-based detectors for Type 1, 2, and 4 variations while showing slightly lower performance for Type 8 variations than the supervised learners (except SVM) with Ext.

TABLE 11. Performance evaluation of profiling-based detection.

Set	Threshold (τ)	TPR	TNR	FPR	FNR	Accuracy	Precision	Recall	F1 score
BASIC	95%	0.439	0.952	0.048	0.561	0.695	0.901	0.439	0.590
	96%	0.421	0.963	0.037	0.579	0.692	0.919	0.421	0.577
	97%	0.400	0.976	0.023	0.600	0.688	0.943	0.400	0.561
	98%	0.391	0.986	0.014	0.609	0.688	0.965	0.391	0.556
	99%	0.349	0.991	0.009	0.651	0.666	0.975	0.349	0.514
EXT	95%	0.997	0.978	0.022	0.003	0.987	0.978	0.997	0.987
	96%	0.996	0.998	0.002	0.004	0.997	0.998	0.996	0.997
	97%	0.996	1.000	0.000	0.004	0.998	1.000	0.996	0.998
	98%	0.994	1.000	0.000	0.006	0.997	1.000	0.994	0.997
	99%	0.993	1.000	0.000	0.007	0.996	1.000	0.993	0.996

TABLE 12. Performance of profiling-based detection against attack variations (Detection rate for Type 4 and average detection rate for Type 1, 2, and 8).

Variation	BASIC	EXT
Type 1	0.930	0.998
Type 2	0.728	1.000
Type 4	1.000	1.000
Type 8	0.777	0.930

Summary: The profiling-based detection captures the characteristics of the Normal instances and then discriminates Spoofed samples deviating from the learned representation. The comparison study shows that the profiling-based detector performs greatly with Ext, while it does not perform well with Basic. The detector with Ext performs comparable or even better than the supervised learning methods, yielding up to 99.8% accuracy for regular attacks and at least 93% detection rates for identifying attack variations.

VI. RELATED WORK

In this section, we summarize previous studies closely related to our study.

The previous work in [17] and [18] tackled the problem of location spoofing in a VANET setting using a set of ML algorithms, including KNN, RF, Naive Bayes, and decision tree. Like our study, the authors take the 2-sequence approach that simply serializes two consecutive BSM messages. The reported performance using the VeReMi dataset is promising, showing over 95% when using KNN and RF across the attack types. However, our evaluation result shows somewhat lower performance, particularly for attack type 8 and 16 (less than 90% using KNN and RF) in our experimental setting.

The work in [16] conducted feature engineering for the detection of location spoofing attacks. The authors established an n-sequence trajectory inspection problem as a generalization of the detection of location spoofing. Three features of Movement Plausibility Check (MPC), Minimum Distance to Trajectories (MDT), and Minimum Translation Distance to Trajectories (MTDT) were defined in this previous work, as discussed in Section III-B. While those features are interesting and helpful for managing detection performance well, they are computationally too heavy to obtain due to the requirement of pair-wise comparison across the training set. In addition, the detection in this previous work requires monitoring three or more advertised messages to expect high-quality prediction. In contrast, our presented scheme performs comparably only with two advertisements without imposing expensive pair-wise computations for feature engineering, which is a crucial element for supporting real-time operations under the provision of restrictive resources common in a mobile communication setting.

A study in [20] defined features for the plausibility check, including MPC (earlier than [16]). The feature of Location Plausibility Check (LPC) predicts the acceptable location boundary based on the current coordinate and velocity information based on 95% and 99% confidence intervals. The authors evaluated the detection performance using KNN and SVM with the two plausibility check features, but the result shows a somewhat lower detection performance than other previous studies.

In [13], the authors defined three features based on interactions between the sender and receiver. The feature of the angle of arrival is defined using the arctangent function with the distance between the sender and receiver. The other two features defined in this previous work are the estimated distance derived from the received signal strength indicator (RSSI) and the pass loss in dB information, and the difference between the estimated and measured distance information. These features were combined with the location and velocity differences for the evaluation performed with conventional ML algorithms, including KNN, RF, and a combination of RF and KNN. The authors in this work assume that the detection takes place at individual vehicles based on the concept of distributed intrusion detection, which is the primary motivation for using the interactions between the sender and receiver. In this study, we do not assume a distributed detection environment.

Additionally, there are some other studies [19], [38], [39] that investigated the location spoofing problem using the VeReMi dataset. While the previous studies reported interesting results, they considered the static attack types defined in the dataset. In this study, we bring up the possibility of attack variations with a set of scenarios defined for potential attack variations. Moreover, we examine the feasibility of zero-day detection for location spoofing using

a profiling-based detection mechanism (often employed in anomaly detection), while the previous research focused on supervised learning with little attention to emerging attack variants.

VII. CONCLUSION

This study presented a data-driven methodology for detecting location spoofing attacks accurately and reliably. In particular, our scheme utilizes a new set of features, which is differential in nature and enables the checking of the mobility constraints and inconsistency. We compared our scheme with the previous research to show that the use of these features improves the detection performance consistently. In this study, we also introduced a set of scenarios to create attack variants with dynamic choices of coordinate offsets when fabricating location information, established in order to evaluate the resilience of the detectors against such variations. The evaluation results confirm the effectiveness of the presented scheme with significantly improved performance across diverse attack variants. To further extend the study of attack variations, we examine the feasibility of the profiling-based approach as a tool for zero-day detection. The VAE-based implementation of the profiling-based detection performs greatly with the newly defined feature set, showing comparable or even better performance than the supervised learning methods often confined to known attacks.

There would be several avenues for future exploration. The profiling-based detection can further be analyzed and optimized, including the architectural structures and the determination of threshold ranges. Another direction is deep learning-based generative models for creating sophisticated attack variants; for example, generative adversarial networks (GANs) have often been applied to create counterfeit samples to deceive the detection process (e.g., intrusion detection). Evading attacks using adversarial attack tools would also be interesting to see their impact on the resilience of the detector function.

REFERENCES

[1] D. Raca, D. Leahy, C. J. Sreenan, and J. J. Quinlan, "Beyond throughput, the next generation: A 5G dataset with channel and context metrics," in *Proc. 11th ACM Multimedia Syst. Conf.*, May 2020, pp. 303–308.

[2] T. F. da Silva Pinheiro, F. A. Silva, I. Fé, S. Kosta, and P. Maciel, "Performance prediction for supporting mobile applications' offloading," *J. Supercomput.*, vol. 74, no. 8, pp. 4060–4103, Aug. 2018.

[3] M. Sadrishojaei, N. J. Navimipour, M. Reshadi, and M. Hosseinzadeh, "A new preventive routing method based on clustering and location prediction in the mobile Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10652–10664, Jul. 2021.

[4] L. Mei, J. Gou, Y. Cai, H. Cao, and Y. Liu, "Realtime mobile bandwidth and handoff predictions in 4G/5G networks," *Comput. Netw.*, vol. 204, Feb. 2022, Art. no. 108736.

[5] T. M. Ho, K.-K. Nguyen, and M. Cheriet, "UAV control for wireless service provisioning in critical demand areas: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 70, no. 7, pp. 7138–7152, Jul. 2021.

[6] S. Sharma, A. Kaul, S. Ahmed, and S. Sharma, "A detailed tutorial survey on VANETs: Emerging architectures, applications, security issues, and solutions," *Int. J. Commun. Syst.*, vol. 34, no. 14, p. e4905, Sep. 2021.

[7] K. Lim, K. M. Tuladhar, and H. Kim, "Detecting location spoofing using ADAS sensors in VANETs," in *Proc. 16th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2019, pp. 1–4.

[8] T. Wang and Y. Yang, "Analysis on perfect location spoofing attacks using beamforming," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 2778–2786.

[9] M. Kamal, A. Barua, C. Vitale, C. Laoudias, and G. Ellinas, "GPS location spoofing attack detection for enhancing the security of autonomous vehicles," in *Proc. IEEE 94th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2021, pp. 1–7.

[10] C. Hu, Y. Liu, Z. Lu, S. Zhao, X. Han, and J. Xiong, "Smartphone location spoofing attack in wireless networks," in *Proc. Int. Conf. Secur. Privacy Commun. Syst.* Cham, Switzerland: Springer, 2021, pp. 295–313.

[11] S. P. Arteaga, L. A. M. Hernandez, G. S. Perez, A. L. S. Orozco, and L. J. G. Villalba, "Analysis of the GPS spoofing vulnerability in the drone 3DR solo," *IEEE Access*, vol. 7, pp. 51782–51789, 2019.

[12] Y. Dang, C. Benzaid, B. Yang, T. Taleb, and Y. Shen, "Deep-ensemble-learning-based GPS spoofing detection for cellular-connected UAVs," *IEEE Internet Things J.*, vol. 9, no. 24, pp. 25068–25085, Dec. 2022.

[13] S. Ercan, M. Ayaida, and N. Messai, "Misbehavior detection for position falsification attacks in VANETs using machine learning," *IEEE Access*, vol. 10, pp. 1893–1904, 2022.

[14] R. W. Heijden, T. Lukaseder, and F. Kargl, "VeReMi: A dataset for comparable evaluation of misbehavior detection in VANETs," in *Proc. Int. Conf. Secur. Privacy Commun. Syst.* Cham, Switzerland: Springer, 2018, pp. 318–337.

[15] J. Kamel, M. Wolf, R. W. van der Hei, A. Kaiser, P. Urien, and F. Kargl, "VeReMi extension: A dataset for comparable evaluation of misbehavior detection in VANETs," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–6.

[16] A. Le and C. Maple, "Shadows don't lie: N-sequence trajectory inspection for misbehaviour detection and classification in VANETs," in *Proc. IEEE 90th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2019, pp. 1–6.

[17] A. Sharma and A. Jaekel, "Machine learning approach for detecting location spoofing in VANET," in *Proc. Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul. 2021, pp. 1–6.

[18] A. Sharma and A. Jaekel, "Machine learning based misbehaviour detection in VANET using consecutive BSM approach," *IEEE Open J. Veh. Technol.*, vol. 3, pp. 1–14, 2022.

[19] S. Gyawali and Y. Qian, "Misbehavior detection using machine learning in vehicular communication networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.

[20] S. So, P. Sharma, and J. Petit, "Integrating plausibility checks and machine learning for misbehavior detection in VANET," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2018, pp. 564–571.

[21] N. S. Rajput, "Measurement of IEEE 802.11 p performance for basic safety messages in vehicular communications," in *Proc. IEEE Int. Conf. Adv. Netw. Telecommun. Syst. (ANTS)*, Dec. 2018, pp. 1–4.

[22] S. Raschka, *Python Machine Learning*. Birmingham, U.K.: Packt, 2015.

[23] C. M. Bishop, *Pattern Recognition and Machine Learning* (Information Science and Statistics). Cham, Switzerland: Springer-Verlag, 2006.

[24] Y. Roh, G. Heo, and S. E. Whang, "A survey on data collection for machine learning: A big data-AI integration perspective," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 4, pp. 1328–1347, Apr. 2019.

[25] S. Gyawali, Y. Qian, and R. Q. Hu, "Machine learning and reputation based misbehavior detection in vehicular communication networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 8871–8885, Aug. 2020.

[26] L. Yang, A. Finamore, F. Jun, and D. Rossi, "Deep learning and zero-day traffic classification: Lessons learned from a commercial-grade dataset," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 4, pp. 4103–4118, Dec. 2021.

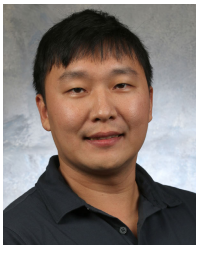
[27] T. Zoppi, A. Ceccarelli, and A. Bondavalli, "Unsupervised algorithms to detect zero-day attacks: Strategy and application," *IEEE Access*, vol. 9, pp. 90603–90615, 2021.

[28] B. I. Hairab, M. S. Elsayed, A. D. Jurcut, and M. A. Azer, "Anomaly detection based on CNN and regularization techniques against zero-day attacks in IoT networks," *IEEE Access*, vol. 10, pp. 98427–98440, 2022.

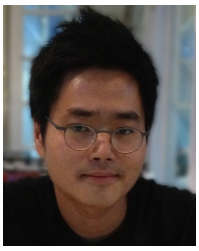
[29] F. Abri, S. Siami-Namini, M. A. Khanghah, F. M. Soltani, and A. S. Namin, "Can machine/deep learning classifiers detect zero-day malware with high accuracy?" in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 3252–3259.

[30] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," *IEEE Trans. Knowl. Data Eng.*, early access, Nov. 23, 2022, doi: 10.1109/TKDE.2021.3130191.
[31] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, Jul. 2009.
[32] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4393–4402.
[33] N. Seliya, A. Abdollah Zadeh, and T. M. Khoshgoftaar, "A literature review on one-class classification and its potential applications in big data," *J. Big Data*, vol. 8, no. 1, pp. 1–31, Dec. 2021.
[34] J. Kim, M. Nakashima, W. Fan, S. Wuthier, X. Zhou, I. Kim, and S.-Y. Chang, "Anomaly detection based on traffic monitoring for secure blockchain networking," in *Proc. IEEE Int. Conf. Blockchain Cryptocurrency (ICBC)*, May 2021, pp. 1–9.
[35] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, San Diego, CA, USA, 2018, pp. 1–15.
[36] Q. P. Nguyen, K. W. Lim, D. M. Divakaran, K. H. Low, and M. C. Chan, "GEE: A gradient-based explainable variational autoencoder for network anomaly detection," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Jun. 2019, pp. 91–99.
[37] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *Special Lect. IE*, vol. 2, pp. 1–18, Dec. 2015.
[38] F. Hawlader, A. Boualouache, S. Faye, and T. Engel, "Intelligent misbehavior detection system for detecting false position attacks in vehicular networks," in *Proc. IEEE Int. Conf. Commun. Workshops*, Jun. 2021, pp. 1–6.
[39] M. A. Elsayed and N. Zincir-Heywood, "BoostGuard: Interpretable misbehavior detection in vehicular communication networks," in *Proc. IEEE/IFIP Netw. Oper. Manage. Symp.*, Apr. 2022, pp. 1–9.

CHIHO KIM (Member, IEEE) is currently pursuing the M.S. degree with the Computer Science Department, Texas A&M University–Commerce. His research interests include security, networking and communications, and machine learning, with a particular interest in detection problems for location spoofing, malware, and network intrusions.



SANG-YOON CHANG (Member, IEEE) received the B.S., M.S., and Ph.D. degrees from the Department of Electrical and Computer Engineering, University of Illinois at Urbana–Champaign, in 2007, 2009, and 2013, respectively. He is currently an Associate Professor with the Computer Science Department, University of Colorado Colorado Springs. Before joining the university, he was a Postdoctoral Fellow with the Advanced Digital Sciences Center. His research interests include security, networking, wireless systems, cyber-physical systems, and applied cryptography.



DONGEUN LEE received the B.S. degree in computer science and engineering and the Ph.D. degree in electrical engineering and computer science from Seoul National University, Seoul, South Korea, in 2006 and 2014, respectively. He was a Computer Systems Engineer with the Lawrence Berkeley National Laboratory and also jointly affiliated as a Postdoctoral Research Associate with the School of Electrical and Computer Engineering, Ulsan National Institute of Science and Technology, Ulsan, South Korea. He joined the Department of Computer Science, Texas A&M University–Commerce, in 2016, where he is currently an Associate Professor. His current research interests include big data analytics for streaming data and scientific machine learning.

JONGHYUN KIM received the Ph.D. degree in computer science from The University of Oklahoma, USA, in 2005. He was a Researcher with Samsung Electronics, from 1995 to 1997. He is currently a Principal Researcher with the Electronics Telecommunications Research Institute (ETRI), Daejeon, South Korea. He is also working as a Project Leader with the Intelligence Security Group, ETRI. He is also involved in standardization activities as the Vice Chair of WP1 and a Rapporteur of Q.4 (cybersecurity) with ITU SG17. His research interests include information security, cyber security, cloud security, AI-based malware detection, and 5G/6G security.

KYUNGMIN PARK received the B.S., M.S., and Ph.D. degrees in computer engineering from Chungnam National University, Republic of Korea, in 2010, 2013, and 2019, respectively. He joined the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Republic of Korea, in 2017, where he is currently working as a Senior Researcher. His research interests include network security and 5G/6G security.

JINOH KIM (Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Minnesota, Twin Cities. He is currently an Associate Professor of computer science with Texas A&M University–Commerce. He is also an Affiliate Faculty Scientist at the Lawrence Berkeley National Laboratory (LBNL) and also a member of the Silicon Valley Cybersecurity Institute (SVCSI). His main research interests include networked systems and distributed computing focusing on performance, reliability, scalability, visibility, and security, using machine intelligence and algorithmic methodologies.