SVM-CASE: An SVM-based Context Aware Security Framework for Vehicular Ad-hoc Networks

Wenjia Li
Department of Computer Science
New York Institute of Technology
New York, New York 10023
Email: wli20@nyit.edu

Anupam Joshi and Tim Finin
Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County (UMBC)
Baltimore, Maryland 21250
Email: {joshi, finin}@cs.umbc.edu

***Abstract*—Vehicular Ad-hoc Networks (VANETs) are known to be very susceptible to various malicious attacks. To detect and mitigate these malicious attacks, many security mechanisms have been studied for VANETs. In this paper, we propose a context aware security framework for VANETs that uses the Support Vector Machine (SVM) algorithm to automatically determine the boundary between malicious nodes and normal ones. Compared to the existing security solutions for VANETs, The proposed framework is more resilient to context changes that are common in VANETs, such as those due to malicious nodes altering their attack patterns over time or rapid changes in environmental factors, such as the motion speed and transmission range. We compare our framework to existing approaches and present evaluation results obtained from simulation studies.**

***Index Terms*—security; malicious attack; context awareness; vehicular ad hoc network; Support Vector Machine**

I. Introduction

A Vehicular Ad-hoc Network (VANET) is normally composed of a *dynamic* set of vehicular nodes that rely on each other to relay packets. VANETs are known to be very vulnerable to various misbehaviors, including malicious attacks and random failures. Therefore, security is always a key challenge for the practical deployment of VANETs.

To detect and mitigate these misbehaviors in VANETs, various security solutions have been studied in the past decade. Most of these security solutions rely on a set of pre- defined and fixed threshold(s) to distinguish misbehaving nodes from normal ones.

However, given the extremely dynamic nature of VANETs, it is not feasible to figure out one set of threshold(s) that work well under all different circumstances. In contrast, we argue that we can apply machine learning techniques, such as the Support Vector Machine (SVM) algorithm, to automatically determine the boundary between the misbehaving nodes and well-behaved nodes.

In addition, we collect and utilize various contextual information to help better distinguish truly malicious nodes from those that conduct abnormal behaviors due to contextual reasons. Figure 1 demonstrates such a good example. We find that node 1 drops packets in both cases, which will make it be treated as misbehaving nodes by most of the existing security mechanisms without any further investigation. However, if we further consider the context with which packet dropping occurs, we find that node 1 drops packets probably because of the busy channel in case (a), and it drops packets purely out of malicious intent in case (b) because there is no external factor that has hindered it from forwarding those packets. Thus, we clearly see from this example that context can play an important role in detecting malicious nodes in VANETs.

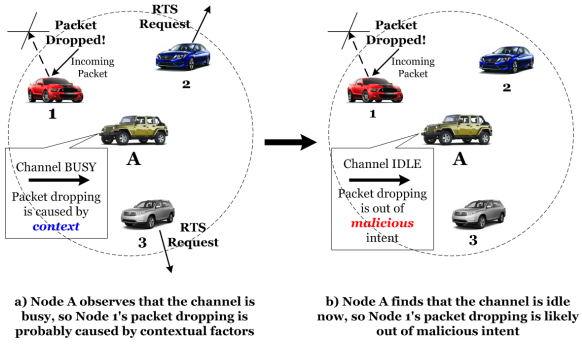


Fig. 1. Context matters when identifying malicious nodes in VANETs

In this paper, we study an SVM-based Context-Aware Security Framework (SVM-CASE) for VANETs. In the SVM-CASE framework, both node behaviors and contextual information are modeled as features in the feature vector to train a SVM classifier. Then the SVM classifier is used to tell whether a node is malicious or not.

This work has the following two main contributions.

- First, we apply the SVM algorithm to automatically distinguish malicious attackers from normal ones rather than relying on the pre-defined and fixed threshold(s). Thus, our security framework is more resilient to the context changes that are common in VANETs, such as (1) those due to malicious nodes altering their attack patterns over time or (2) rapid changes in environmental factors, such as the motion speed and transmission range.
- Second, we model the contextual information (such as velocity, temperature, altitude, etc.) as features in the feature vector for the SVM algorithm. By this means, we can detect malicious nodes in a more accurate manner without manually defining complicated security policies to utilize the contextual information.

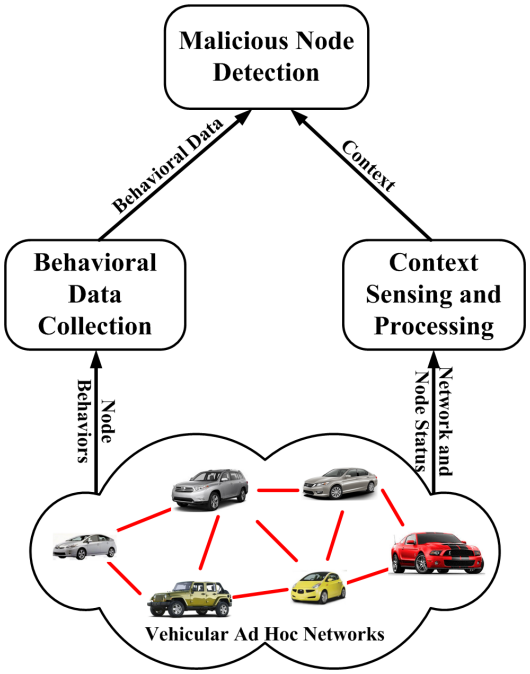


Fig. 2. The SVM-CASE Framework

II. Related Work

In the past decade, some research efforts have been made to detect and mitigate misbehaviors for ad hoc networks.

Intrusion Detection System (IDS) has been viewed as an important solution for detecting various misbehaviors in ad hoc networks. Several approaches have been proposed to build IDS probes on each mobile node due to the lack of a fixed infrastructure, such as [1]. On the other hand, Huang et al. [2] proposed a cooperative intrusion detection framework, in which clusters are formed and a node in each cluster will act as the cluster head in turn and coordinate amongst all the cluster members for the intrusion detection process.

In VANETs, there have been some research efforts to detect misbehaviors. Raya et al. [3] studied a protocol to identify misbehaviors in vehicular networks using clustering technique. In [4], Markov model was used to identify false alerts reported by misbehaving vehicles. Ruj et al. proposed a data-centric misbehavior detection method for VANETs [5]. However, these approaches only deal with misbehaviors in general, and they cannot further distinguish malicious attacks from random errors or faulty behaviors.

III. The SVM-CASE Framework

In this section, we will describe the SVM-CASE framework in details. The SVM-CASE framework has three functional modules: *Behavior Data Collection*, *Context Sensing and Processing*, and *Misbehavior Detection*, , and the framework is illustrated in Figure 2.

A. Behavioral Data Collection

The behavioral data collection module is responsible for the collection of node behaviors and formation of behavioral dataset. In this paper, a node's behavior is described in terms of the ratio of the amount of this behavior over the total amount of packets that the node has received, such as *packet drop rate* (PDR), *packet modification rate* (PMR) and *RTS flooding rate* (RTS).

We use network simulation to generate behavioral dataset that is used to train a SVM classifier. Because the adversaries and their misbehaviors are pre-defined in these simulations, the behavioral data are collected and then labeled according to the ground truth regarding the adversaries. The trained SVM classifier can then be distributed and deployed to vehicular nodes to distinguish malicious nodes from normal ones.

During the testing stage, the Behavioral Data Collection module on each node first observes and records the behaviors of their neighbors. It also receives and integrates node behaviors reported by its *immediate* neighbors. (i.e., nodes that are within its radio range.) In this paper, we use Dempster-Shafer Theory [6] to combine the observations, as discussed in details in Section III-D.

B. Context Sensing and Processing

As we show in Figure 1, contextual factors sometimes can hinder a node from behaving normally. Thus, we sense and record a set of contextual information in the proposed framework so that they can be used together with nodes' own behaviors to determine if a node is malicious or not. There are several types of contextual information that can significantly impact a node's behaviors, such as velocity, channel status, temperature, GPS coordinates, altitude, and weather, etc. The effect of each contextual information on nodes' behaviors is analyzed as follows.

- ***Velocity*** indicates a vehicle's movement, and we observe that the faster a vehicle travels, the more difficult it is to cooperate with other nodes.
- ***Channel status*** reflects how busy the channel is during a specific period of time. Because all the nodes within the radio range share the transmission channel, data packets are more likely to get dropped if the channel get busier. The higher the value, the more busier the channel.
- ***Temperature and wind speed*** are both important factors to determine whether or not a node's misbehaviors are caused by harsh weather or not. For instance, a mobile node is more likely to malfunction in a case when the temperature is $20°F$ and the wind speed is $40mph$ than a case when the temperature is $70°F$ and the wind speed is $5mph$.
- ***GPS coordinates and altitude*** together indicate the geolocation of the mobile nodes, which can then be used to determine if a node is forced to conduct misbehaviors due to its location. For example, if a mobile node travels to the opposite side of a hill from its current communication peer, then it has to drop the packet because the hill may have blocked the communication channel.

From the analysis above, we find that all these contextual factors can help us better understand the causes of misbehaviors. By integrating the contextual factors into the malicious

node detection process, we can identify the malicious nodes in a more accurate manner.

In addition, it is also very feasible to collect these contextual information due to the wide deployment of smartphones, such as Apple iPhone and Android phone. Both types of smartphones are equipped with various sensors, such as accelerometer, GPS receiver, temperature sensor, etc. Thus, many of the contextual factors listed above can be easily collected by smartphones in practice.

Since the range of values of the *raw* contextual data varies widely, the range of all contextual data should be rescaled so that each of them contributes approximately equally to the SVM classifier. In our framework, all the contextual data will be rescaled to the range of $[0, 1]$.

The rescaled contextual data and the node behaviors are both used as features in the feature vector of the SVM algorithm. An example of the training data is shown in Table I.

We can tell from Table I that Node 1 is a normal node which conducts misbehaviors (packet dropping and modification) because of its high velocity. On the other hand, Node 2, 3 and 4 are all malicious because their misbehaviors are NOT caused by environmental factors. Node 5 is not malicious because its misbehaviors are caused by very busy channel.

C. Malicious Node Detection

The support vector machine (SVM) algorithm [7] is used in our framework to detect malicious nodes in VANETs.

Initially, each node observes and records neighbor behaviors, and these local observations are fed into the SVM classifier to produce the initial classification result. Because each node only observes behaviors of its direct neighbors, the local observations are then exchanged among nodes so that each node can also know the behaviors of other nodes that are out of its radio range. The local observations and external observations obtained from other nodes are fused together using Dempster-Shafer Theory and thus an updated behavioral dataset is generated. If the updated behavioral dataset makes the SVM classifier produce a different classification result, then the updated behavioral dataset is propagated to all neighbors. Once there is not any change in the classification result when they receive foreign behavioral data, the procedure terminates. At this point, all the nodes have the same belief of malicious nodes.

D. Observation Combination

Observation combination is one of the most important functionalities in our proposed framework. Because some of the incoming observations are not reliable, it is essential to find a combination technique to properly fuse together multiple pieces of observations.

Dempster-Shafer theory of evidence (DST) is known to be an appropriate technique to fuse together multiple piece of observations even if some of them might not be accurate. In DST, probability is replaced by an uncertainty interval bounded by belief (bel) and plausibility (pls). Belief is the lower bound of this interval and represents supporting evidence. Plausibility is the upper bound of the interval and represents non-refuting evidence. For instance, if a node N_k observes that one of its neighbors, say node N_j, has dropped packets with probability p, then node N_k has p degree of belief in the packet dropping behavior of node N_j and 0 degree of belief in its absence. The belief value with respect to an event α_i and observed by node N_k can be computed as the following.

$$bel_{N_k}(\alpha_i) = \sum_{e:\alpha_e \in \alpha_i} m_{N_k}(\alpha_e)$$

Here α_e are all the basic events that compose the event α_i, and $m_{N_k}(\alpha_e)$ stands for the view of the event α_e by node N_k. In this case, since node N_k merely get one single report of node N_j from itself, i.e., $\alpha_i \subset \alpha_i$. Therefore, we can derive that $bel_{N_k}(\alpha_i) = m_{N_k}(\alpha_i)$. Note that $\bar{\alpha_i}$ denotes the non-occurrence of the event α_i. Since the equation $pls(\alpha_i) = 1 - bel(\bar{\alpha_i})$ holds for belief and plausibility, we can further derive the following: $bel_{N_k}(N_j) = m_{N_k}(N_j) = p$ and $pls_{N_k}(N_j) = 1 - bel_{N_k}(\bar{N_j}) = 1$.

Given that belief indicates the lower bound of the uncertainty interval and represents supportive evidence, we define the combined packet dropping level of node N_j as the following.

$$pd_{N_j} = bel(N_j) = m(N_j) = \bigoplus_{k=1}^{K} m_{N_k}(N_j)$$

Here $m_{N_k}(N_j)$ denotes the view of node N_k on another node N_j. We can combine reports from different nodes by applying the Dempster's rule, which is defined as following.

$$m_1(N_j) \bigoplus m_2(N_j) = \frac{\sum_{q,r:\alpha_q \cap \alpha_r = N_j} m_1(\alpha_q) m_2(\alpha_r)}{1 - \sum_{q,r:\alpha_q \cap \alpha_r = \Phi} m_1(\alpha_q) m_2(\alpha_r)}$$

IV. Performance Evaluation

In this section, we evaluate the performance of the SVM-CASE framework using network simulation, and its performance is compared to two baseline mechanisms. The first baseline mechanism is the Context-Aware Security and Trust Framework (*CAST*) proposed in [8], and the second baseline mechanism is the weighted voting (WV) mechanism, which has been widely used in detecting misbehaviors in wireless networks [9], [10].

A. Adversary Model

In this paper, we assume that the adversary can conduct multiple types of misbehavior at the same time, and it can mix these misbehaviors at any ratio. In addition, the adversary can alter this ratio over time. For instance, an adversary A may determine at time t_1 that it should equally conduct the three types of misbehaviors (i.e., RTS flooding, packet dropping, and packet modification); while at time t_2, A changes its misbehavior pattern to solely perform RTS flooding attack.

B. Simulation Setup

We use GloMoSim 2.03 [11] as the experimental platform, and table II lists the parameters used in the simulation scenarios.

$Node\ ID$	$Drop$	$Modify$	RTS	$Channel$	$Velocity$	$Temperature$	$Malicious?$
1	0.8	0.2	0	0.1	**0.9**	0.5	No
2	0	0.5	0.5	0.1	0.2	0.4	Yes
3	0.3	0.3	0.4	0.1	0.1	0.6	Yes
4	0.2	0.1	0.7	0.1	0.2	0.4	Yes
5	0.9	0	0.1	**0.9**	0.1	0.6	No

TABLE I
AN EXAMPLE OF TRAINING DATASET OF NODE BEHAVIORS AND CONTEXTUAL DATA

$Parameter$	$Value$
Simulation area	$600m \times 600m$
Num. of nodes	50, 100, 200
Transmission range	$120m$
Node motion speed	$5m/s, 10m/s, 20m/s, 30m/s$
Num. of bad nodes	5, 10, 15, 20, 25, 30, 35, 40
Simulation time	900s

TABLE II
SIMULATION PARAMETERS

$Node\ ID$	$Start$	End	$Drop$	$Modify$	RTS
15	$0s$	$900s$	0.1	0.9	0.0
23	$0s$	$900s$	0	0.4	0.6
31	$0s$	$900s$	0.4	0.4	0.2
34	$0s$	$900s$	0.2	0.1	0.7
42	$0s$	$900s$	0.3	0	0.7

TABLE III
AN EXAMPLE OF MISBEHAVIOR SETUP FOR TESTING STAGE

We use three parameters to evaluate the correctness and efficiency of the SVM-CASE framework: *Precision*, *Recall*, and *Communication Overhead* (CO). P, R, and CO are defined as follows.

$$P = \frac{Num\ of\ Truly\ Malicious\ Devices\ Caught}{Total\ Num\ of\ Untrustworthy\ Devices\ Caught} \quad (1)$$

$$R = \frac{Num\ of\ Truly\ Malicious\ Devices\ Caught}{Total\ Num\ of\ Truly\ Malicious\ Devices} \quad (2)$$

$$CO = \frac{Number\ of\ Packets\ for\ Detecting\ Misbehaviors}{Total\ Number\ of\ Packets\ in\ VANETs}$$

The experiments are conducted in two phases: the training stage and the testing stage. In the training stage, there are totally 100 nodes in VANET, with five of them being misbehaving nodes and the rest 95 nodes being well-behaved nodes. These five misbehaving nodes conduct a mixed set of various misbehaviors including packet dropping, packet modification, RTS flooding, and fake observation propagation, and the mixture rate for these misbehaviors varies among the misbehaving nodes. However, the mixture rate for each misbehaving node is fixed throughout the training stage. Table III depicts an example setup of misbehaviors for our simulation. Note that *Start* and *End* denote the start time and end time for the specified set of misbehaviors respectively. Each value in the columns of *Drop*, *Modify* and *RTS* stands for the percentage of the corresponding misbehavior over the whole set of misbehaviors in the specified time range, and the percentages of these three misbehaviors sum up to one. Just take the first entry as an example: 80% of misbehaviors conducted by node 1 will be packet dropping and 20% of that be packet modification during the time range between 0s and 900s.

In the testing stage, we vary the number of misbehaving nodes in different experimental scenarios, not only the mixture rates for different nodes vary, the mixture rate for the same misbehaving node conducts also differs over time. In this way, we ensure that the training datasets are significantly different from the testing datasets.

Each simulation scenario has 30 runs with different random seeds, which ensures a unique initial node placement for each run. Each experimental result is the average over the 30 runs for this simulation scenario.

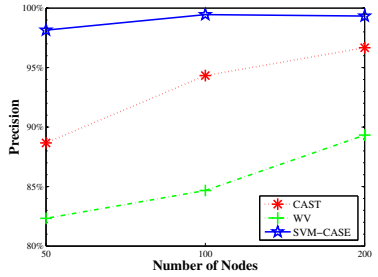
C. Evaluation Result

We observe the performance of *SVM-CASE* against those of *CAST* and *WV* in the following three scenarios: different number of nodes, different percentage of misbehaving nodes, and different node motion speeds. Given that the simulation area remains the same in all these scenarios, we can observe from these scenarios the effect of node density, adversary percentage, and node mobility, respectively. The simulation results are showed in the following Figure 3, Figure 4, and Figure 5.

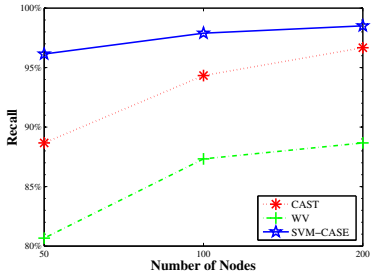
From Figure 3, Figure 4, and Figure 5, we find that the SVM-CASE outperforms the two baseline mechanisms because it yields higher precision and recall values with a reasonable communication overhead.

V. CONCLUSION

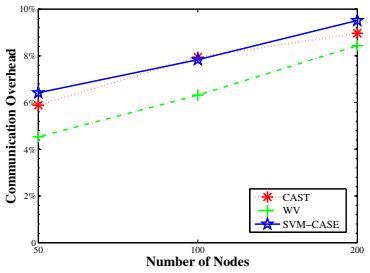
In this paper, we describe an SVM-based Context-Aware Security Framework (SVM-CASE) for VANETs. We use both behaviors observed by neighboring nodes and a set of contextual factors to train a SVM classifier, and then use the SVM classifier to distinguish misbehaving nodes from well-behaved nodes. Experimental results show that the SVM-CASE framework achieves a good performance in terms of high precision and recall values and acceptable communication overhead.



(a) Precision

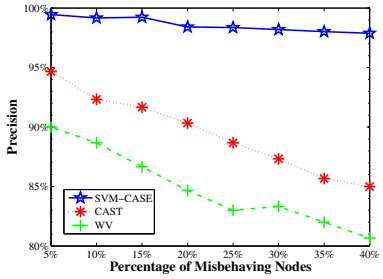


(b) Recall

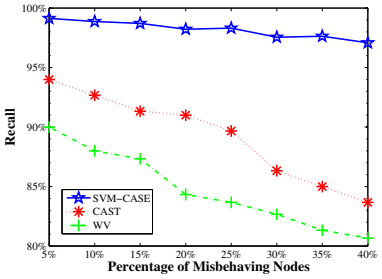


(c) Communication Overhead

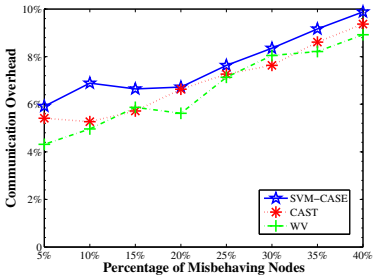
Fig. 3. Effect of Node Density



(a) Precision

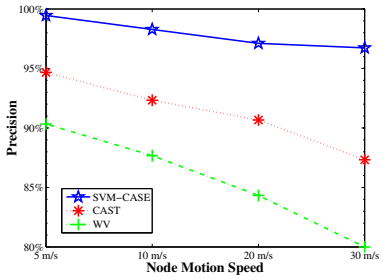


(b) Recall

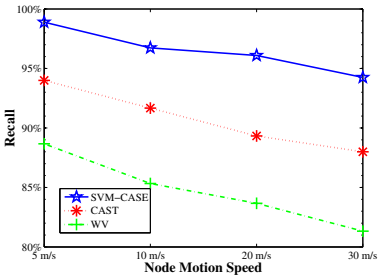


(c) Communication Overhead

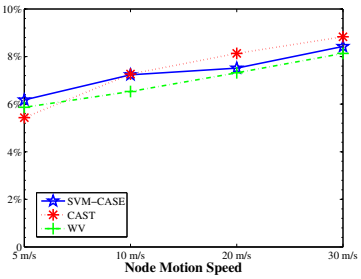
Fig. 4. Effect of Adversary Percentage



(a) Precision



(b) Recall



(c) Communication Overhead

Fig. 5. Effect of Node Mobility

REFERENCES

[1] Y. Zhang and W. Lee, "Intrusion detection in wireless ad-hoc networks," in *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2000, pp. 275–283.

[2] Y.-A. Huang and W. Lee, "A cooperative intrusion detection system for ad hoc networks," in *SASN '03: Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*. New York, NY, USA: ACM, 2003, pp. 135–147.

[3] M. Raya, P. Papadimitratos, I. Aad, D. Jungels, and J.-P. Hubaux, "Eviction of misbehaving and faulty nodes in vehicular networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 8, pp. 1557–1568, Oct 2007.

[4] M. Ghosh, A. Varghese, A. Kherani, and A. Gupta, "Distributed misbehavior detection in vanets," in *IEEE Wireless Communications and Networking Conference, 2009. WCNC 2009.*, April 2009, pp. 1–6.

[5] S. Ruj, M. Cavenaghi, Z. Huang, A. Nayak, and I. Stojmenovic, "On data-centric misbehavior detection in vanets," in *2011 IEEE Vehicular Technology Conference (VTC Fall 2011)*, Sept 2011, pp. 1–5.

[6] G. Shafer, *A Mathematical Theory of Evidence*. Princeton, NJ, USA: Princeton University Press, 1976.

[7] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines : and other kernel-based learning methods*, 1st ed. Cambridge University Press, March 2000.

[8] W. Li, A. Joshi, and T. Finin, "Cast: Context-aware security and trust framework for mobile ad-hoc networks using policies," *Distributed and Parallel Databases*, vol. 31, no. 2, pp. 353–376, 2013.

[9] J. Parker, A. Patwardhan, and A. Joshi, "Cross-layer analysis for detecting wireless misbehavior," in *Proceedings of the Third IEEE Consumer Communications and Networking Conference, 2006. CCNC 2006.*, vol. 1. IEEE, Jan. 2006, pp. 6–9.

[10] I.-R. Chen, F. Bao, M. Chang, and J.-H. Cho, "Dynamic trust management for delay tolerant networks and its application to secure routing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 5, pp. 1200–1210, May 2014.

[11] X. Zeng, R. Bagrodia, and M. Gerla, "Glomosim: a library for parallel simulation of large-scale wireless networks," *ACM SIGSIM Simulation Digest*, vol. 28, no. 1, pp. 154–161, 1998.