BOTVEILLANCE: A Vehicular Botnet Surveillance Attack against Pseudonymous Systems in VANETs

Mevlut Turker Garip, Peter Reiher, Mario Gerla
Department of Computer Science, University of California Los Angeles
{mtgarip, reiher, gerla}@cs.ucla.edu

***Abstract*—Vehicular ad hoc networks (VANETs) use inter-vehicle communications to improve traffic safety by broadcasting information such as vehicle speed, location and heading to other vehicles. This approach depends on each vehicle advertising its location information. Since the pseudonyms (identifiers) of the vehicles are also broadcast, attackers can track any vehicle of interest, violating privacy of the drivers. The most widely accepted defense is continuous pseudonym updates. We present BOTVEILLANCE, an adaptive cooperative surveillance attack by vehicular botnets—effective even against the best existing pseudonym changing scheme. It is the first long-range global-scale surveillance attack that is solely performed by vehicles themselves without depending on any additional hardware. Since we use our vehicular bots, our surveillance attack is not confined to a specific area. We show via simulation that our attack can keep a vehicle under surveillance up to 85 percent of its route, and identify its destination address 90 percent of the time.**

***Index Terms*—Vehicular Ad Hoc Networks, VANET Security, Vehicular Botnets, Surveillance Attack, Tracking, Pseudonym Changing Schemes, Pseudonymous Systems, Location Privacy**

I. Introduction

Many traffic accidents are caused by unsafe driver actions due to insufficient traffic information [27]. Vehicular ad hoc networks (VANETs) can enhance traffic information accuracy by having vehicles exchange traffic information through Basic Safety Messages (BSMs) [30], which contain current speeds, locations, directions, etc. The direct activation of commands (brakes, steering wheel, etc.) by an alarm will allow prompt reaction to abrupt traffic events. VANET traffic safety applications rely on the location data broadcast in BSMs, allowing an attacker to track vehicles, violating driver privacy.

Location privacy is a crucial security requirement for VANETs. Confidential information such as the home address of a driver can be identified by analyzing the routes taken by the vehicle [14], which can be obtained by a surveillance attack and used for malign purposes. The only realistic and most widely accepted method to provide location privacy in VANETs—given their requirement of low-latency communication—is the use of changing pseudonyms. They are abstract identifiers embedded in certificates and BSMs. They must be included in each BSM to provide authentication and non-repudiation [30], uniquely identifying a vehicle without revealing the identity of its driver unless requested by authorities per the *accountability requirement* of pseudonyms.

When a vehicle changes its pseudonym, all the other uniquely identifying information in the BSM must be changed as well. No pseudonym can be reused because valid old pseudonyms permit Sybil attacks with multiple valid identities [20]. As a result, since the pseudonyms are short-lived and the issuance of each pseudonym takes more time than low-latency VANETs can tolerate, vehicles locally store a *pseudonym set* containing the next pseudonyms to be used. After this set is depleted, a new one is obtained from the authorities.

A pseudonym changing defense's strategy for how, where and in what kind of situations pseudonyms should be changed is a challenging research problem [2]. A simple pseudonym changing scheme cannot prevent tracking. The Swing pseudonym changing scheme [16] is widely regarded as best in terms of effectiveness, feasibility and robustness.

In this paper, we present BOTVEILLANCE, the first surveillance attack in the literature that can perform long-range global-scale tracking without any additional hardware, road-side equipment or visual contact. We achieve this by taking advantage of powerful adversarial entities—vehicular botnets. Since methods for compromising a single vehicle have been demonstrated in [21] and [23], we show in [10] that compromising multiple vehicles and organizing them into botnets is very feasible. Vehicular botnets allow powerful attacks that are impossible with a single compromised vehicle. BOTVEILLANCE uses geographically scattered vehicular bots to track the victim car. They cooperate with each other using GHOST [12], our secret vehicular botnet communication concealed inside BSM broadcasts. This cooperation enables the tracker bots to cover more grounds with their combined communication ranges, making long-range global-scale tracking possible. Based on the position and mobility of the tracked car and vehicular bots, the tracking task and history are handed over from bot to bot, each being selected in a distributed manner, making our attack adaptive to unexpected conditions and mobility patterns. BOTVEILLANCE is effective against most pseudonym changing schemes; we only demonstrate its efficacy against the Swing scheme. We exploit immutable vehicle properties and location prediction heuristics to link pseudonyms. Since vehicles might advertise ambiguous location data to prevent tracking [7], vehicular bots use INTERLOC [11], our interference-resistant RSSI-based localization mechanism, which can accurately locate the tracked car using the signal strength of its broadcasts.

In Section II, we survey existing pseudonym changing schemes and discuss related work on tracking attacks. In Section III, we present the design details of BOTVEILLANCE

ISBN 978-3-903176-03-4

and the description and configuration of Swing. In Section IV, we discuss the accuracy metrics chosen for the evaluation of the attack, and show its effectiveness via an extensive and thorough experimentation. In Section V, we suggest possible solutions to improve location privacy for VANETs. In Section VI, we conclude by discussing the contributions of our attack to the future research on location privacy in VANETs.

II. Related Work

A. Pseudonym Changing Schemes

Location privacy in VANETs is necessary to ensure broad acceptance and deployment of these networks. An effective pseudonym changing scheme is the key to defending against tracking attacks. Thus, many pseudonym changing schemes have been proposed, which can be split into six groups [20]:

Fixed Change Period: Vehicles change their pseudonyms in a fixed period [8]. Attackers can easily link the old and new pseudonyms and track the victim by figuring out the period.

Random Change Period: Vehicles change their pseudonyms in random unpredictable periods [18]. Since this makes multiple vehicles changing their pseudonyms at the same time rare, it is still easy to track the victim by mobility analysis.

Silent Periods: This strategy is not an alternative to the others but rather a supplement. Vehicles stop broadcasting any message for a chosen period of time after pseudonym change [14] [22] to make it harder to link the old and new pseudonyms by mobility analysis. However, being silent at critical places such as intersections endanger traffic safety. We believe that the best method for unpredictable silent periods without endangering traffic safety is randomly setting them and waking up when necessary for traffic safety.

Density-Based Change: If the victim car is the only one that changes its pseudonym, an attacker can simply link the old and new pseudonyms by filtering out the pseudonyms of other vehicles that do not change. If multiple cars change their pseudonyms but they are far from each other, linking is still possible by mobility analysis. Therefore, an ideal scheme would aim for multiple vehicles in close proximity to perform simultaneous change. In density-based pseudonym changing schemes [5] [6], a vehicle changes its pseudonym when the neighbor vehicles are above a density threshold. Yet, sometimes none of the neighbors might want to change pseudonyms to prolong the lifespan of their *pseudonym set*. Also, when traffic is sparse, this scheme would force the victim not to change its pseudonym for a long time.

Collaborative Change: [9] proposes applying the concept of *mix-zone* to VANETs for location privacy. Mix-zones are regions where every BSM exchanged is encrypted with the corresponding mix-zone key and can only be decrypted by members of that mix-zone. This way, an attacker trying to track a vehicle outside his mix-zone would not be able to decrypt any of the victim's BSMs to obtain location information or to detect its pseudonym change [2] [13] [25]. However, the mix-zone defense is vulnerable to an insider attack. Given that each mix-zone has to be large enough for a sufficient traffic awareness, the probability of a vehicular bot being located in victim car's mix-zone is significantly high.

Vehicle-Centric Change: In these schemes, each vehicle is free to choose the best strategy for its current situation. We believe that this group of schemes are the most effective and realistic since the same strategy applied in every situation will not provide everyone with the optimal privacy. Two examples are presented in [16]: Swap and Swing. In Swap, neighbors change their pseudonyms by exchanging them with each other with 50% chance and enter a random silent period. However, this causes many issues with accountability, and enables Sybil attacks if the old pseudonyms stay active after the exchange. Swing, on the other hand, performs these collaborative pseudonym changes without causing new security issues. We believe that it is the most effective pseudonym changing scheme in terms of its capabilities, feasibility and robustness. It has all the necessary features of an effective pseudonym changing scheme while being realistic and light-weighted (unlike mix-zones with constant encryption) and without compromising traffic safety. Swing, which we use to evaluate our attack, is described in more detail in Section III-C.

B. Tracking Attacks

The design of tracking attacks that can target these various schemes has not received enough attention. The existing work either does not describe the implementation details of such attacks, or has unrealistic requirements and assumptions.

[17] presents a tracking attack that creates trajectory predictions through a matrix completion algorithm. However, the attack is not tested against any pseudonym changing scheme. [4] also does not take any scheme into account. Also, it assumes one all-seeing attacker without explaining how a single vehicle can obtain complete knowledge of the network.

Before we discuss the next set of attacks, the term *global-scale tracking attack* has to be defined: the attack that can be performed anywhere at any time for any duration on a victim vehicle with an arbitrary route of an undetermined distance.

[1] and [19] depends on installation of sniffing stations for tracking in a confined local area where it is realistic to deploy them, making it not a global-scale tracking attack. [26] suggests using compromised road side units (RSUs) for linking victim's pseudonyms and constructing its past mobility trace. However, whereas the methods for compromising a vehicle are demonstrated in [21] and [23], RSUs may not be universally deployed and it is uncertain if they could be compromised. Even if they could, using stationary RSUs to track a mobile victim would not be as effective as using mobile trackers like our vehicular bots. Any tracking attack that depends on additional stationary hardware has be confined to a local area. Our attack is the first global-scale surveillance attack without the need for any additional hardware—evaluated to be effective even against the strongest pseudonym changing strategies.

[15], [22] and [29] mention global passive adversaries without any suggestions on how they can be implemented. [13] describes a simple tracking attack by a global adversary where only mobility analysis is used for context-linking. Its tracking fails if there is more than one estimated tracked car.

III. BOTVEILLANCE

A. Overview

Here we describe BOTVEILLANCE's major design elements—the immutable vehicle properties and location prediction heuristics to link pseudonyms, the mechanism to choose the best tracker bot for adapting to changing conditions, and how it achieves the long-range global-scale tracking without requiring additional hardware. We also describe Swing as a defense and its configurations to evaluate our attack. The design details of our other mechanisms, INTERLOC and GHOST, are provided in [11] and [12] respectively. GHOST provides the secret communication infrastructure needed for vehicular bots to coordinate their attacks. INTERLOC correlates location broadcasts from the tracked car.

B. Identifiers to Change Beside Pseudonyms

BSMs contain information in addition to pseudonyms that could also be used as tracking identifiers by attackers. Therefore, it is widely accepted that these identifiers (MAC address, IP address, etc.) must be changed along with the vehicle's pseudonym. The size of a vehicle, which is constantly advertised in BSMs, is not considered to be among these identifiers since it does not uniquely identify the vehicle. Even though this is an accurate assumption for a large area with numerous cars, we show that it could be exploited as an identifier if the area where the tracked car is being searched is small. The vehicle size information in BSMs has a fine granularity (in centimeters) [30], and each vehicle model has a unique size in centimeters; having two cars with the same make and model in an area as small as the estimation algorithm of our attack can calculate is unlikely. We exploit this observation as a heuristic for linking the old and new pseudonyms of the tracked car. Vehicle size cannot be omitted from BSMs as a defense since it is a vital piece of information for traffic safety.

	Percentage (%)	Width (cm)	Length (cm)
Ford F-Series	4.69	203	532
Chevrolet Silverado	3.29	203	523
Ram P/U	2.80	202	531
Toyota Camry	2.22	183	485
Honda Civic	2.10	180	450
Toyota Corolla	2.06	178	464
Honda CR-V	2.04	183	455
Toyota RAV-4	2.01	185	460
Honda Accord	1.97	185	483
Nissan Rogue	1.89	183	462

Figure 1. Top 10 most popular car models in the US with their dimensions

The table in Figure 1 shows the top 10 most popular car models in the US with their percentages and dimensions [3]. Even though widths can be the same for some models due to the approximation to the closest centimeter value, the dimensions of a vehicle as a pair of $(width, length)$ is unique.

We implement the car models and dimensions in Figure 1 for the evaluation. The brand of each car in simulation is chosen randomly—with the probability proportional to its popularity—to test our attack heuristic with real-life data.

C. Swing

Swing includes all the essential features for an effective pseudonym changing scheme, such as collaborative pseudonym change, random pseudonym change period and random silent period. To better describe how Swing works, we first describe its parameters:

Minimum Pseudonym Change Period ($period_{min}$): The minimum time before pseudonym change is allowed, trading off privacy level and scalability of pseudonym issuance.

Maximum Pseudonym Change Period ($period_{max}$): The lifespan of each pseudonym, after which it must change.

Minimum Silent Period ($silent_{min}$): The minimum silent period required after each pseudonym change.

Maximum Silent Period ($silent_{max}$): The maximum silent period allowed after each pseudonym change. Since being silent at critical places such as intersections endangers traffic safety, we wake vehicles up at those places, possibly sooner than $silent_{max}$.

Maximum Neighbor Distance ($neighbor_{max}$): The maximum distance allowed between two neighboring vehicles for collaborative pseudonym change to occur. A greater distance may not provide sufficient attacker confusion.

$period_{min}$	$period_{max}$	$silent_{min}$	$silent_{max}$	$neighbor_{max}$
5 seconds	10 seconds	2 seconds	10 seconds	100 meters

Figure 2. Values of the Swing parameters chosen to test BOTVEILLANCE

Figure 2 shows the values of the Swing parameters for the evaluation of BOTVEILLANCE. These values are chosen to favor privacy—rather than pseudonym issuance scalability or traffic safety—in order to subject our attack to rigorous testing.

In Swing, after a vehicle changes its pseudonym, it broadcasts a *Pseudonym Change Notification* (CNG_N) to neighbors within $neighbor_{max}$, and waits for $period_{min}$ before it can perform another change. After $period_{min}$, vehicles change their pseudonyms only if they receive a CNG_N from any neighbor in the vicinity of $neighbor_{max}$ for collaborative pseudonym change. If a vehicle cannot perform a collaborative change for $period_{max}$, it will change its pseudonym and broadcast a CNG_N if there is at least one neighbor in the vicinity of $neighbor_{max}$ regardless of whether a CNG_N is received from it or not. If there is no such neighbor, it will wait for a second and check again. When the neighbors receive a CNG_N, only the ones with pseudonyms that are $period_{min}$ or older will join the collaborative change; the others will ignore the CNG_N. This makes prediction of the time of next pseudonym change infeasible. Also, the independent and distributed nature of pseudonym change decisions makes this scheme more light-weight and robust to changing conditions.

After pseudonym change, each vehicle enters a silent period randomly chosen between $silent_{min}$ and $silent_{max}$. The time passed in silence does not reduce the lifespan of the next pseudonym, that is, counting towards $period_{min}$ does not start until the silent period ends. All of these mechanisms together provide a significant level of confusion against attackers.

D. BOTVEILLANCE Initialization

Our attack begins with the botmaster ordering vehicular bots to track the victim car. The order message must contain the current pseudonym of the victim or some other clearly identifying information. The communication between the botmaster and vehicular bots can be performed either over the Internet or using GHOST if the botmaster has a vehicle. For the more frequent communication among vehicular bots during the attack, GHOST is used to avoid detection.

Once the order is received by all the bots, they start the search for the car with the target pseudonym in their communication range using the BSMs received. Each bot that locates the victim car sends a *Tracking Response Packet* (RES_{TR}) to all other bots and the botmaster to announce its candidacy as the tracker bot. If the botmaster does not receive a RES_{TR}, it will keep retransmitting the order until at least one tracker bot is found. The content of RES_{TR} is as follows:

Estimation Error ($error_{est}$): Let pos_{est} be the position of the tracked car after the silent period—estimated by mobility analysis. Let pos_{dim} be the position of the vehicle that is closest to pos_{est} among all the vehicles in the bot's communication range with the same dimensions as the tracked car. $error_{est}$ is the distance between pos_{est} and pos_{dim}. $error_{est}$ is not used for the first tracker bot selection since no estimation is required—the victim car's pseudonym is already known.

Distance to the Tracked Car (d_{tr}): The distance between the tracker bot candidate and the tracked car.

Pseudonym of the Candidate ($pseudo_{cand}$): The pseudonym of the tracker bot candidate sending the RES_{TR}.

During RES_{TR} broadcasts from the candidates, bots that could not locate the victim car ignore these messages. These RES_{TR} exchanges determine who wins the competition to be the first tracker bot. Each bot checks if its d_{tr} is smaller than the d_{tr} in every RES_{TR} received from the other candidates. If not, the bot will cease its tracking. In the rare cases that the d_{tr}s of two candidates are equal, comparison between the $pseudo_{cand}$s will break the tie, ensuring in a distributed manner that only one bot will continue tracking. This is the initial tracker bot selection; continuous selections and tracking history handoffs are explained in the subsequent section.

E. BOTVEILLANCE Attack

1) Tracker Bot's Attack State: After the initial tracker bot is selected, it initializes its *tracking state* ($state_{tr}$) and begin tracking the victim. $state_{tr}$, a compound data structure, stores the necessary information about the tracked car and current tracking state. Any bot receiving it can resume tracking from where it left off. The content of this $state_{tr}$ data structure is:

Tracked Car Information ($info_{tracked}$): The content of the last BSM heard from the tracked car, constantly updated with each new BSM. It is used to estimate the position of the tracked car when it is in silent period. It contains the tracked car's last heard pseudonym, speed, direction, dimensions, latitude and longitude, and the timestamp of the BSM ($last_{heard}$).

Estimated Velocity of the Tracked Car in Silence (v_{est}): The estimated velocity of the tracked car during the silent period. v_{est} is constantly updated with the moving average of the velocities (speeds and directions) advertised in the BSMs of the tracked car and all the vehicles in front of it.

After initialization, the tracker bot begins tracking the victim and recording each advertised position in a database ($history_{tr}$). The tracker bot keeps the $state_{tr}$—$info_{tracked}$ and v_{est}—updated at all times with the incoming BSMs.

2) Continuous Calculations of the Estimation Area: Figure 3 shows the tracking mechanism in progress and estimation area around the pos_{est}, which moves based on the value of v_{est} and $last_{heard}$. v_{est} is calculated and constantly updated even while the tracked car is not in silent period so that the tracker bot has an accurate v_{est} whenever the tracked car's random pseudonym change occurs. Since v_{est} continues to be updated with the moving average of the velocities of the vehicles in front of the tracked car while it is in silent period, our attack can still accurately estimate the tracked car's mobility during silent period because it is constrained by their mobility pattern.

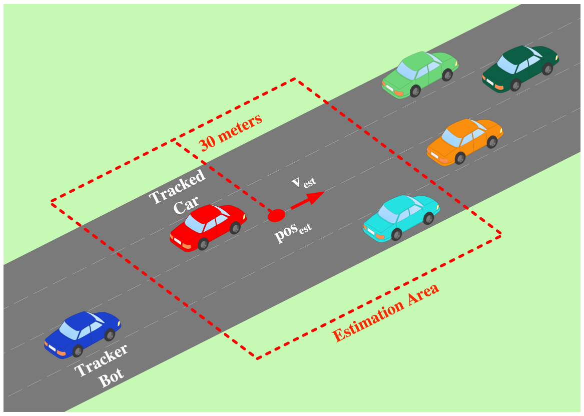

Figure 3. Tracking mechanism, v_{est} calculation and moving estimation area

The pos_{est}—the estimated position of the tracked car—is calculated by adding the multiplication of the v_{est} and time elapsed since $last_{heard}$ to the tracked car's latitude and longitude in the $info_{tracked}$ according to the direction of the v_{est}. This estimation is continuously performed to ensure responsive context-linking once the silent period is detected. While the tracked car's BSMs are being received without any break, the pos_{est} will overlap with the actual position of the tracked car because the time elapsed since $last_{heard}$ will be almost zero. For each calculated pos_{est}, an estimation area will be created with the pos_{est} at the center to compensate for estimation errors. Although the width of a highway in one direction is around 18 meters (5 lanes x 3.6 meters [28]), we set the radius of the estimation area to 30 meters to be robust to the ongoing turns and/or GPS errors of the tracked car. Every time estimation area is updated based on pos_{est}, the tracker bot checks if the time elapsed since $last_{heard}$ has become $silent_{min}$ to detect silent periods. At least $silent_{min}$ has to pass without receiving any BSM from the tracked car before the tracker bot attempts context-linking, since any duration less than that might be caused by packet losses. The continuous v_{est} and pos_{est} updates protect the tracker bot from the adverse effects of this wait on the tracking accuracy.

3) Context-Linking Mechanism: After $silent_{min}$, the tracker bot begins its context-linking attempts to link the old and new pseudonyms of the tracked car. It constantly monitors the BSMs to see if any are sent from within the estimation area by a vehicle with the same dimensions as the tracked car. If not, the context-linking attempt will be repeated when the estimation area is updated. Since the v_{est}, pos_{est} and corresponding estimation area updates are as frequent as BSM broadcasts ($\approx 100\ ms$), the context-linking attempts will be just as frequent. If any of the attempts locates such a vehicle inside the estimation area, the context-linking will be complete, and the old pseudonym of the tracked car and the pseudonym of this vehicle will be linked. If there are multiple cars with the same dimensions as the tracked car in the estimation area, then the tracker bot will consider the one that is closest to the pos_{est} as the tracked car. The estimation area radius is chosen for better context-linking accuracy: large enough to contain the tracked car despite estimation errors, but small enough to minimize the number of vehicles with the tracked car's dimensions. Finally, after the context-linking is complete, the $state_{tr}$ update mechanism will be configured with the tracked car's new pseudonym so that it can be updated with the correct data. Also, the number of velocity samples so far for the v_{est} calculation will be reset to one so that the velocities advertised in the future can have a bigger impact on the moving average than the old ones.

4) Continuous Tracker Bot Selections: In case context-linking keeps failing, the attempts will continue only until $silent_{max}$ after $last_{heard}$ because then it would mean that the tracked car is lost since the silent period cannot be longer than $silent_{max}$. Afterwards, the tracker bot will give up, initiate the next tracker bot selection and perform the $history_{tr}$ handoff.

Tracker bot selection starts with the current tracker bot calculating all the possible pos_{est}s. For every intersection the tracked car might have passed, the number of possible pos_{est}s will increase. The tracker bot will search its communication range for the vehicles with the same dimensions as the tracked car. In case of multiple such vehicles being found, the tracker bot—like all other tracker bot candidates—has to determine the most likely tracked car. It will be the closest one to any one of the calculated pos_{est}s—providing the smallest $error_{est}$ as defined in Section III-D. Finally, the tracker bot will calculate the $error_{est}$ for and its distance (d_{tr}) to the most likely tracked car, and send a *Tracking Request Packet* (REQ_{TR}) to all the other bots to check if any of them is a tracker bot candidate.

REQ_{TR} has the same content as RES_{TR}—$error_{est}$, d_{tr} and $pseudo_{cand}$—except it also has the tracker bot's $state_{tr}$ so that the other bots can perform the search for the tracked car. If the tracker bot could not locate any vehicle with the same dimensions as the tracked car in its communication range, it will set the $error_{est}$ and d_{tr} to a very high value in its REQ_{TR}. The other bots that receive this REQ_{TR} will initialize their $state_{tr}$ and calculate the corresponding possible pos_{est}s. They will search for the tracked car in their communication range and go through the same $error_{est}$ and d_{tr} calculations for the most likely tracked car. However, the bots that could not locate any such vehicle—instead of setting a very high $error_{est}$ and d_{tr}—will just ignore the REQ_{TR}. Afterwards, the tracker bot candidates which were able to locate at least one possible tracked car will broadcast a RES_{TR}, entering the competition to be the next tracker bot.

For each received RES_{TR}, every candidate including the current tracker bot will compare its $error_{est}$ with the $error_{est}$ advertised in the RES_{TR} to see if its $error_{est}$ is smaller. If even one received RES_{TR} has a smaller $error_{est}$, the candidate will withdraw from the competition. If two $error_{est}$s are equal, the smaller d_{tr} will win. If the d_{tr}s of the candidates that have the same $error_{est}$ are also equal, the comparison between their unique $pseudo_{cand}$s will determine the winner.

Using $error_{est}$ as the primary factor in choosing the next tracker bot ensures that the vehicle that has the highest probability of being the tracked car is found inside the collective communication range of all the bots. If the current tracker bot determines that it is not the best candidate, it will send its whole $history_{tr}$ to the botmaster, which will append it to the tracking traces sent by the previous tracker bots. During the selection, if the tracker bot does not receive a RES_{TR} from any bot and cannot locate any possible tracked car itself, it will keep sending the REQ_{TR} until there is at least one potential tracker bot. After the selection, the next tracker bot will reset the number of velocity samples for the v_{est} to one to favor the velocities advertised in the future in the v_{est} calculation.

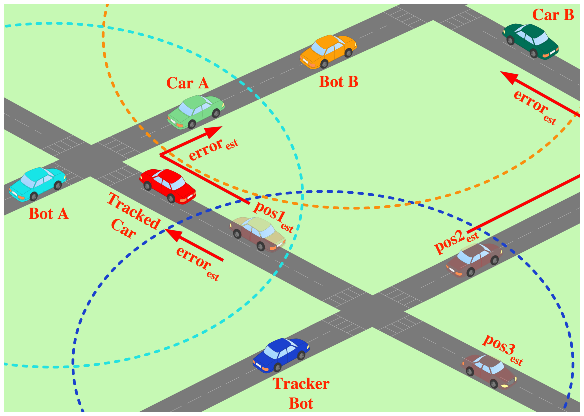

Figure 4. Calculation of multiple pos_{est}s due to an intersection and the tracker bot selection using the $error_{est}$s according to these pos_{est}s

Figure 4 shows a tracker bot selection scenario with multiple possible pos_{est}s due to the tracked car passing an intersection—$pos1_{est}$, $pos2_{est}$ and $pos3_{est}$. It also shows how the $error_{est}$s are used to choose the most likely tracked car. In this scenario, there are two cars with the same dimensions as the tracked car—Car A and Car B—and none of them is in the tracker bot's communication range. Therefore, the tracker bot sends a REQ_{TR} with the highest $error_{est}$ and d_{tr}. Upon receipt of this REQ_{TR}, Bot A locates the tracked car and Car A, while Bot B locates Car A and Car B. Bot A uses the $pos1_{est}$ for calculating both $error_{est}$s since the $pos1_{est}$ is the closest to both the tracked car and Car A. While Bot B uses the $pos1_{est}$ to calculate the $error_{est}$ for Car A, it uses the $pos2_{est}$ for Car B since it is closer. Based on these calculations, Bot A chooses the tracked car whereas Bot B chooses Car A as the most likely tracked car, minimizing their $error_{est}$s. After the RES_{TR}s are exchanged, Bot A becomes the next tracker bot due to having the smallest $error_{est}$ among all the candidates.

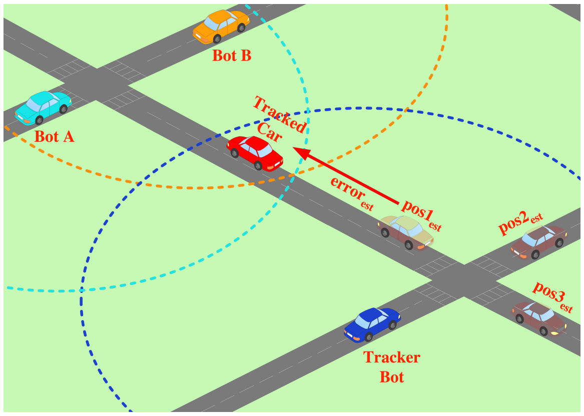

Figure 5. Tracker bot selection where the d_{tr}s determine the next tracker bot

In Figure 5, all the tracker bot candidates have the same $error_{est}$. In this scenario, the tracker bot has the tracked car in its communication range too. Therefore, it is also a candidate and sends a REQ_{TR} with its actual $error_{est}$ and d_{tr} values. Given that all the candidates have the same $error_{est}$ for the most likely tracked car, Bot A becomes the next tracker bot since it has the smallest d_{tr}. This scenario also shows that the continuous tracker bot selections ensure that the best tracker bot performs the tracking most of the time. For example, even though the tracker bot has the tracked car in its communication range, the tracked car is about to leave it; using d_{tr} as a factor in the tracker bot selection hands off tracking to Bot A, which is likely to keep the tracked car under surveillance longer.

5) Edge Cases and Limitations: When the tracker bot is about to leave the map, it must terminate tracking, during either the tracker bot selection or context-linking attempts. It will send its $history_{tr}$ to the botmaster, along with the REQ_{TR} with the highest $error_{est}$ and d_{tr}, to be broadcast to every bot near all possible pos_{est}s. These bots will broadcast their RES_{TR} to the other bots and to the botmaster. If the botmaster does not receive any RES_{TR}, it will keep sending the REQ_{TR} until a bot claims the tracker bot responsibility.

In situations where there is not even a single tracker bot candidate in the vicinity of the tracked car, there will be a gap in the $history_{tr}$. However, since the v_{est} somewhat reflects the average speeds of the roads in the vicinity most of the time, in many cases in our simulations the tracked car was located by a bot through mobility analysis some time after it was lost. We observed that the probability of another vehicle with the same dimensions as the tracked car being located at one of the possible pos_{est}s—constantly updated since the tracked car is lost—is low. That is how our attack always has a reasonable tracking accuracy even in the harshest conditions.

IV. Evaluation

We used Veins [24] (which combines the SUMO and OMNeT simulators) to evaluate BOTVEILLANCE. SUMO is responsible for simulating realistic vehicular traffic while OMNeT simulates the IEEE 802.11p standards [30].

We subjected BOTVEILLANCE to rigorous and challenging experiments. Swing parameters are set to favor privacy. We used a Manhattan mobility model to maximize the number of intersections and possible turns to create the tough conditions for a tracker. We ran 150 simulations in total, each of which was 2 hours long. Each simulation created a total of 3600 cars, providing more than sufficient neighbor density for Swing to achieve the desired privacy levels with frequent collaborative pseudonym changes. During each simulation, the tracked car traveled a total of ≈ 70 kilometers. Given that 10 simulations are run for each value on the x-axis of each graph, two accuracy metrics to evaluate our attack are defined as follows:

Percentage of Route: Let per_{tr} be the percentage of the tracked car's route that the vehicular bots were able to track. The accuracy metric *percentage of route* is then calculated by averaging all the per_{tr}s from 10 simulations.

Destination Detection Accuracy: This accuracy metric is calculated by checking—out of 10 simulations—how many simulations the vehicular bots were able to determine the destination address of the tracked car.

Each graph shows tracking accuracy when varying an attack or Swing configuration, while keeping all the other configurations at their standard values, which are:

Vehicular Bot Percentage	Communication Range	Estimation Area Radius	Pseudonym Change Frequency	Minimum Silent Period
20%	300 meters	30 meters	10 seconds	2 seconds

Figure 6. Standard values for the varied Swing and attack configurations

These values are chosen for realism based on VANET standards, traffic safety and pseudonym issuance scalability, rather than optimal tracking accuracy. Each graph uses more challenging values to test the robustness of our attack and show the relation between each configuration and tracking accuracy.

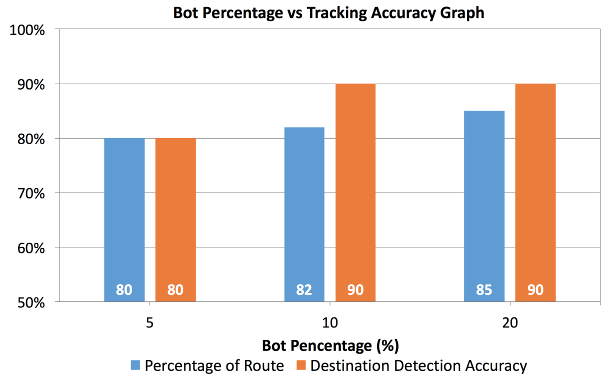

Figure 7. Tracking accuracies with different percentages of vehicular bots

There is a direct correlation between the tracking accuracy and size of the area that is covered by the collective communication range of vehicular bots. Figure 7 shows the tracking accuracy for each percentage of vehicular bots over all the cars in the vicinity of the tracked car along its route during simulation—not the overall percentage of vehicular bots. The tracking accuracy increases linearly with the vehicular bot percentage since the covered area grows linearly with the number of bots. The slope of change in the tracking accuracy, however, is small; that is, a significant decline in the number of vehicular bots causes only a small drop on the tracking accuracy. The is because using d_{tr} as a factor in the tracker bot selections minimizes the number of tracker bots needed during the simulation by maximizing the probability of each one keeping the tracked car under surveillance for a long time.

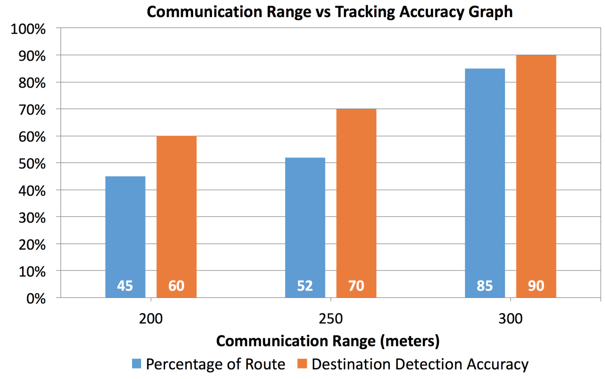

Figure 8. Tracking accuracies with different wireless communication ranges

Figure 8 shows the tracking accuracy for each vehicle wireless communication range. The standard range for IEEE 802.11p is 300 meters [30]. We evaluate BOTVEILLANCE at smaller ranges since the effective communication range of a vehicle might shrink due to environmental factors like interference and obstacles. The results in Figure 8 reaffirm the direct correlation between the tracking accuracy and size of the area covered by vehicular bots. The tracking accuracy changes exponentially with communication range since the covered area is proportional to the communication range squared.

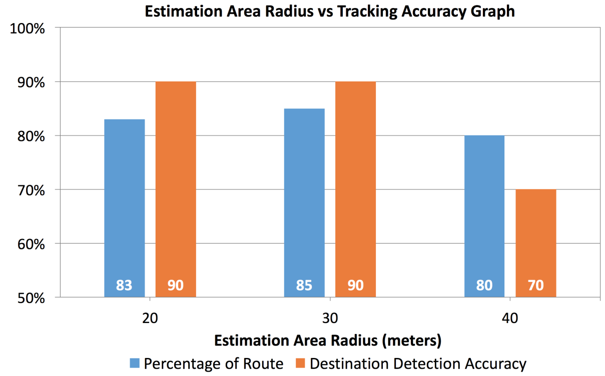

Figure 9. Tracking accuracies with estimation areas of different radii

The radius of the estimation area is also a factor affecting tracking accuracy, particularly context-linking accuracy. A suboptimal radius increases the probability that either the tracked car is lost after the pseudonym change or the tracker bot ends up tracking another car with the same dimensions as the tracked car. Figure 9 shows the tracking accuracy for each radius chosen for the estimation area. The accuracy moves on a concave line with the peak value at the optimal radius. The value of this radius is chosen based on road characteristics.

Swing parameters have significant impact on the tracking accuracy due to the frequency and level of confusion they cause in the tracker bots. We investigate only the effects of two parameters—pseudonym change frequency ($period_{max}$) and minimum silent period ($silent_{min}$)—on the tracking accuracy since the other Swing parameters have similar effects.

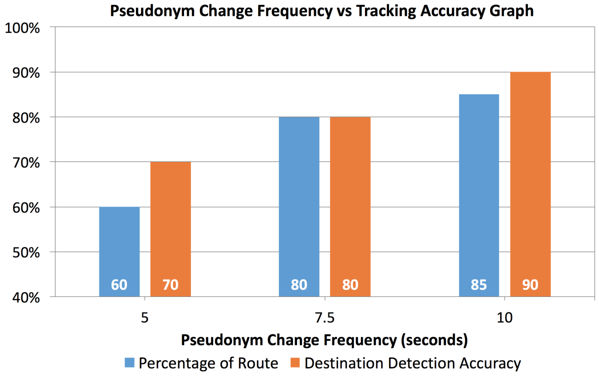

Figure 10. Tracking accuracies with different pseudonym change frequencies

Figure 10 shows the tracking accuracy for each $period_{max}$. The value of $period_{max}$ for our attack, 10 seconds, is already borderline unrealistic due to pseudonym issuance scalability. However, we still investigate the robustness of our attack with lower $period_{max}$ values in case of extreme scenarios with high vehicle density where most vehicles change their pseudonyms collaboratively between $period_{min}$ and $period_{max}$.

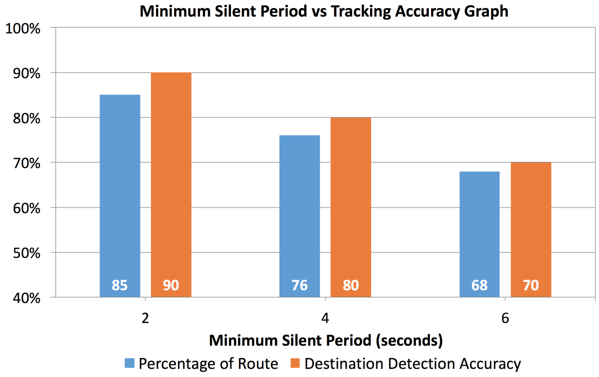

Figure 11. Tracking accuracies with different minimum silent periods

Silent periods challenge tracking significantly; however, their values are limited by traffic safety requirements. Figure 11 shows the tracking accuracy for each $silent_{min}$, which is a lower bound for random silent period selections. Regardless of how unrealistically high the $silent_{min}$ is, our attack achieves a reasonable tracking accuracy due to its effective context-linking heuristics and continuous tracker bot selections.

V. Possible Countermeasures

We showed that vehicle size can be an identifier, so it must be changed along with the pseudonym, but without impacting VANET safety distance calculations. Each vehicle could advertise dimensions within a range greater than its actual dimensions, which would complicate context-linking, as long as each calculated safety distance is acceptable. Alternatively, the centimeter granularity of the size field in BSMs could be decreased—reducing variation in car size.

Attackers may use mobility analysis for context-linking during the silent periods after pseudonym changes, so each vehicle could use a random mobility pattern—which is unnaturally different than the current common mobility pattern on the road—before entering the silent period, increasing the probability of context-linking to the wrong vehicle.

Silent periods limit tracking accuracy, but at a cost. Long silent periods or silence in places like intersections can be unsafe. A special broadcasting mechanism could ensure traffic safety without vehicles ending their silent periods prematurely. Vehicles could send encrypted BSMs directly to an RSU while in silent period. The RSU could then decrypt the BSMs and broadcast them on behalf of the vehicles after removing their pseudonyms. The anonymized BSMs could not be used for tracking, while traffic safety could be ensured.

Using one pseudonym for a group of vehicles rather than an individual vehicle complicates context-linking in tracking attacks. While the other tracking attacks in Section II fail in this case, our attack can still track a vehicle, albeit with less accuracy. If there is another vehicle with the same dimensions as the tracked car in the same group, during a silent period, they can switch positions, which might cause attackers to start tracking the wrong vehicle. However, providing accountability for group pseudonyms is very challenging.

VI. Conclusion

In this paper, we presented BOTVEILLANCE—an adaptive cooperative surveillance attack performed by vehicular botnets. It is the first long-range global-scale tracking attack in the literature without requiring any additional hardware. It is designed with an exhaustive consideration of the most effective pseudonym changing strategies. We discussed the existing pseudonym changing schemes and standards. By demonstrating the powerful adversaries, vehicular botnets, we widened the attack surface for future researchers so that effective and all-inclusive defenses can be designed. We showed via thorough experimentation that BOTVEILLANCE can track a vehicle along 85 percent of its route and detect its destination address 90 percent of the time. We proposed possible countermeasures, which include the improvements on the existing location privacy solutions for pseudonymous systems and on the VANET standards.

VII. Acknowledgement

We would like to thank Paul Hyungmin Kim for his great work contributing to the implementation of the pseudonym changing scheme Swing, and for his valuable feedback.

References

[1] I. Bilogrevic, I. Aad, P. Ginzboorg, V. Niemi, and J.-P. Hubaux. Track me if you can: On the effectiveness of context-based identifier changes in deployed mobile networks. In *NDSS*, 2012.

[2] L. Buttyán, T. Holczer, and I. Vajda. On the effectiveness of changing pseudonyms to provide location privacy in vanets. In *ESAS*, 2007.

[3] T. Cain. 2016 year end u.s. vehicle sales rankings. http://www.goodcarbadcar.net/2017/01/usa-2016-vehicle-sales-by-model-manufacturer-brand.html, 2017.

[4] S. Chapkin, B. Bako, F. Kargl, and E. Schoch. Location tracking attack in ad hoc networks based on topology information. In *IEEE MASS*, 2006.

[5] B. Chaurasia, S. Verma, G. Tomar, and A. Abraham. Optimizing pseudonym updation in vehicular ad-hoc networks. *Transactions on Computational Science IV*, pages 136–148, 2009.

[6] B. K. Chaurasia, S. Verma, G. S. Tomar, and S. Bhaskar. Pseudonym based mechanism for sustaining privacy in vanets. In *IEEE CICSYN*, 2009.

[7] M. Duckham and L. Kulik. Simulation of obfuscation and negotiation for location privacy. In *COSIT*, 2005.

[8] D. Eckhoff, C. Sommer, T. Gansen, R. German, and F. Dressler. Strong and affordable location privacy in vanets: Identity diffusion using time-slots and swapping. In *IEEE VNC*, 2010.

[9] J. Freudiger, M. Raya, M. Félegyházi, P. Papadimitratos, and J. P. Hubaux. Mix-zones for location privacy in vehicular networks. In *ACM WiN-ITS*, 2007.

[10] M. T. Garip, M. E. Gursoy, P. Reiher, and M. Gerla. Congestion attacks to autonomous cars using vehicular botnets. In *NDSS*, 2015.

[11] M. T. Garip, P. H. Kim, P. Reiher, and M. Gerla. Interloc: An interference-aware rssi-based localization and sybil attack detection mechanism for vehicular ad hoc networks. In *IEEE CCNC*, 2017.

[12] M. T. Garip, P. Reiher, and M. Gerla. Ghost: Concealing vehicular botnet communication in the vanet control channel. In *IEEE IWCMC*, 2016.

[13] M. Gerlach and F. Guttler. Privacy in vanets using changing pseudonyms-ideal and real. In *IEEE VTC*, 2007.

[14] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady. Achieving guaranteed anonymity in gps traces via uncertainty-aware path cloaking. *IEEE Transactions on Mobile Computing*, 9(8):1089–1107, 2010.

[15] L. Huang, H. Yamane, K. Matsuura, and K. Sezaki. Towards modeling wireless location privacy. In *International Workshop on Privacy Enhancing Technologies*, 2005.

[16] M. Li, K. Sampigethaya, L. Huang, and R. Poovendran. Swing & swap: user-centric approaches towards maximizing location privacy. In *ACM Workshop on Privacy in Electronic Society*, 2006.

[17] C. Lin, K. Liu, B. Xu, J. Deng, C. W. Yu, and G. Wu. Vclt: An accurate trajectory tracking attack based on crowdsourcing in vanets. In *ICA3PP*, 2015.

[18] Y. Pan, J. Li, L. Feng, and B. Xu. An analytical model for random changing pseudonyms scheme in vanets. In *IEEE NCIS*, 2011.

[19] J. Petit, D. Broekhuis, M. Feiri, and F. Kargl. Connected vehicles: Surveillance threat and mitigation. In *Black Hat Europe*, 2015.

[20] J. Petit, F. Schaub, M. Feiri, and F. Kargl. Pseudonym schemes in vehicular networks: A survey. *IEEE Communications Surveys & Tutorials*, 17(1):228–255, 2015.

[21] S. Rosenblatt. Car hacking code released at defcon. http://news.cnet.com/8301-1009_3-57596847-83/car-hacking-code-released-at-defcon/, 2013.

[22] K. Sampigethaya, M. Li, L. Huang, and R. Poovendran. Amoeba: Robust location privacy scheme for vanet. *IEEE JSAC*, 25(8), 2007.

[23] G. Smith. Driverless car could be hacked by 14-year-old from indonesia, senator warns. http://www.huffingtonpost.com/2013/05/17/driverless-car-hack_n_3292748.html, 2013.

[24] C. Sommer. Veins: Vehicles in network simulation. http://veins.car2x.org, 2015.

[25] A. Studer, E. Shi, F. Bai, and A. Perrig. Tacking together efficient authentication, revocation, and privacy in vanets. In *IEEE SECON*, 2009.

[26] Y. Sun, R. Lu, X. Lin, X. Shen, and J. Su. An efficient pseudonymous authentication scheme with strong privacy preservation for vehicular communications. *IEEE TVT*, 59(7):3589–3603, 2010.

[27] U.S. Department of Transportation. National motor vehicle crash causation survey. *DOT HS 811 059*, 2008.

[28] U.S. Department of Transportation FHWA. Mitigation strategies for design exceptions. http://safety.fhwa.dot.gov/geometric/pubs/mitigation strategies/chapter3/3_lanewidth.cfm, 2014.

[29] M. Wernke, P. Skvortsov, F. Dürr, and K. Rothermel. A classification of location privacy attacks and approaches. *JPUC*, 18(1):163–175, 2014.

[30] Wireless LAN Working Group. Wireless access in vehicular environments. *IEEE Standards*, July 2010.