UNIVERSITY OF CALIFORNIA

Los Angeles

Design and Mitigation of Vehicular Botnets in Vehicular Ad Hoc Networks

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Computer Science

by

Mevlut Turker Garip

2019

ABSTRACT OF THE DISSERTATION

Design and Mitigation of Vehicular Botnets in Vehicular Ad Hoc Networks

by

Mevlut Turker Garip
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2019
Professor Peter L. Reiher, Co-Chair
Professor Leonard Kleinrock, Co-Chair

Improving traffic safety is considered as one of the most important tasks by many countries, and current solutions are not very effective due to the human factor. There are 30,000 deaths and 2.2 million injuries caused by car accidents each year just in the US alone. Many of these accidents are due to the lack of sufficient traffic information and slow driver reaction. Vehicular ad hoc networks (VANETs) is a peer-to-peer communication protocol designed to improve traffic safety. In order to achieve this, cars communicate with each other over a wireless channel exchanging information such as their current speeds, locations, heading angles, brake system statuses, etc. VANETs help collect traffic information more accurately and from a much wider area than drivers.

Autonomous driving eliminates the human factor, which will prevent traffic accidents more effectively. Autonomous vehicles are expected to be on the market in 2020. VANET will be used by these cars since it decreases the dependency on expensive sensors, thus making the deployment of autonomous vehicles easier and faster. The accurate traffic information collected using VANETs from a wide area will provide highly effective collision avoidance tools in these autonomous vehicles. However, these benefits are counterbalanced by possible security attacks. Researchers showed that computers inside these vehicles can be compromised; as a result, attackers can take full control of them. Similar to Internet botnets, we can expect that attackers will organize these vehicles into vehicular botnets. We demonstrate the particular dangers of this threat by designing vehicular botnets that consist of these compromised autonomous cars. Existing research into security problems with autonomous vehicles cannot properly cope with our vehicular botnets since they fail to consider the possibility of cooperative malicious cars.

This dissertation investigates the potential uses of vehicular botnets, how they can be organized and controlled, and how we might detect and defend against them. We discuss how an attacker who controls a number of vehicles can organize them into a botnet using the current VANET infrastructure. We demonstrate three powerful attacks performed by our vehicular botnets. We then present our detection mechanism to find vehicular botnets, discuss possible approaches to evict them from the network, and the effect of their eviction. Our work is the first that ever proposed and designed vehicular botnets in the literature. The evaluation of our work is solely through simulations since experimenting with real autonomous vehicles is impractical and dangerous.

The dissertation of Mevlut Turker Garip is approved.

Douglas S. Parker

Lixia Zhang

Leonard Kleinrock, Committee Co-Chair

Peter L. Reiher, Committee Co-Chair

University of California, Los Angeles

2019

To my parents, for their continual love and support

and for teaching me how to be honest and responsible

and to my sisters, for their encouragement and motivation

and to the memory of my dearly beloved grandmother, Ipek.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I would like to pay my tributes to Professor Mario Gerla, who has recently passed away, for his contributions to my dissertation work, especially for coming up with the idea of vehicular botnets in the first place. I would also like to express my thanks and appreciation to my committee members—Professor Kleinrock, Professor Zhang, and Professor Parker—for taking time to serve on my committee and for providing me with insightful feedback.

I would like to give a very special thanks to my advisor, Professor Peter Reiher, for his supervision throughout my graduate career. At every stage of my dissertation work, his valuable advice always led me in the right direction. He never minded the long research meetings—in fact, he encouraged them—and was always interested in helping me solve even the most detailed design challenges. His continuous patience and support made it possible for me to overcome any type of problem during my PhD studies and complete my dissertation work. He was and remains a great role model for a scientist and mentor.

I would like to acknowledge the contributions of some researchers whom I had the opportunity to collaborate with. I would like to thank Deepak Ananth Rama, Mehmet Emre Gursoy, Paul Hyungmin Kim, Amir Saad, and Jonathan Lin for contributing to different aspects of my dissertation work. The papers on RIoT and SHIELDNET—where Amir and Jonathan helped me, respectively—are currently in the submission process.

I would also like to thank Lauren Samy for all the times she went to great lengths to help me with whatever problems I have encountered during my PhD studies; for her everlasting positive energy, encouragement, and moral support whenever I felt troubled; and for continuing to be an amazing and dependable friend. I would like to thank Yasen Yasenov for being a loyal and close friend, for being the reason I was able to avoid culture shock in the beginning of my PhD studies, and for always being there and able to comfort me whenever I needed him.

Finally, I would like to express my gratitude to my parents and my sisters for their continuous love, support and encouragement, which helped me get to where I am today. I would also like to thank my grandmother, who has recently passed away, for being a role model for a kind, sacrificing and incredibly positive person.

VITA

2010-2011	Research Assistant, Computer Security Laboratory, University of California, Davis. Worked on Buffer Overflow, SQL Injection, XSS and Cross Site Forgery Attacks.
2012	B.S. (Computer Science), Bilkent University, Ankara, Turkey.
2012-2013	Research Assistant, Laboratory for Advanced Systems Research, University of California, Los Angeles. Worked on the prevention of MITM attacks in communications with implantable wireless medical devices. Designed and proved a novel defense against MITM attacks called selective jamming.
2016	M.S. (Computer Science), University of California, Los Angeles.
2013-2018	Teaching Assistant/Associate/Fellow, Computer Science Department, University of California, Los Angeles. Taught sections of CS 31 (introductory programming course) under direction of David Smallberg. Taught sections of CS 35L (introductory software construction course) under direction of Professor Paul Eggert. Taught CS 111 (operating systems principles course), CS 136 (computer security course), and CS 188 (distributed systems course) under direction of Professor Peter Reiher.

PUBLICATIONS

Mevlut Turker Garip, Peter Reiher, and Mario Gerla. BOTVEILLANCE: A Vehicular Botnet Surveillance Attack against Pseudonymous Systems in VANETs. In IFIP Wireless and Mobile Networking Conference (WMNC) (pp. 1-8), 2018.

Zhongliang Zhao, Eryk Schiller, Eirini Kalogeiton, Torsten Braun, Burkhard Stiller, Mevlut Turker Garip, Joshua Joy, Mario Gerla, Nabeel Akhtar, and Ibrahim Matta. Autonomic Communications in Software-driven Networks. IEEE Journal on Selected Areas in Communications (JSAC), 35(11):2431-2445, 2017.

Mevlut Turker Garip, Paul Hyungmin Kim, Peter Reiher, and Mario Gerla. INTERLOC: An Interference-aware RSSI-based Localization and Sybil Attack Detection Mechanism for Vehicular Ad Hoc Networks. In IEEE Consumer Communications & Networking Conference (CCNC) (pp. 1-6), 2017.

Mevlut Turker Garip, Peter Reiher, and Mario Gerla. GHOST: Concealing Vehicular Botnet Communication in the VANET Control Channel. In IEEE International Wireless Communications and Mobile Computing Conference (IWCMC) (pp. 1-6), 2016.

Mevlut Turker Garip, Mehmet Emre Gursoy, Peter Reiher, and Mario Gerla. Congestion Attacks to Autonomous Cars Using Vehicular Botnets. In Network and Distributed System Security Symposium (NDSS), 2015.

Mevlut Turker Garip, Mehmet Emre Gursoy, Peter Reiher, and Mario Gerla. Scalable Reactive Vehicle-to-Vehicle Congestion Avoidance Mechanism. In IEEE Consumer Communications & Networking Conference (CCNC) (pp. 943-948), 2015.

Victor Perez, Mevlut Turker Garip, Silas Lam, and Lixia Zhang. Security Evaluation of a Control System Using Named Data Networking. In IEEE International Conference on Network Protocols (ICNP) (pp. 1-6), 2013.

CHAPTER 1

Introduction

1.1 Traffic Safety

Traffic safety is a main concern for many countries [OGZ97]. Countries apply new traffic regulations and take many precautions on the infrastructure, but the improvement rates are still not promising; therefore, utilizing the advantages of modern technologies to prevent accidents is considered to be the most effective approach [Eva03]. Many accidents are caused by insufficient traffic information and by slow driver reaction/attention to local visual and acoustic inputs [KGS10]. How soon is enough for a driver to react to prevent an accident? Is the range of sight a safe distance to begin taking actions? How about during severe weather conditions which will worsen the range of sight? VANETs have been proposed to solve these problems and provide better collision avoidance mechanisms. They overcome these problems by enhancing both the accuracy of traffic information and the delivery of alarms, thus helping prevent collisions. In a VANET, cars communicate with each other over a wireless channel. They can send packets directly to neighbors if they are within radio range. Otherwise, intermediary cars act like routers and forward the packets to intended destinations based on a routing protocol deployed in the network. Communication among the vehicles is performed in a peer-to-peer fashion, without any centralized coordination. In VANETs, cars can exchange routine information such as current speeds, locations, directions, as well as emergency alarms like notifications of emergency braking, etc. With VANETs, cars can collect more accurate traffic information electronically than drivers can visually. Using vehicular networking for collision avoidance and direct activation of commands (brakes, accelerator, steering wheel, etc.) by an alarm will ensure a car's prompt reaction without depending on the driver's alertness.

1.2 VANET

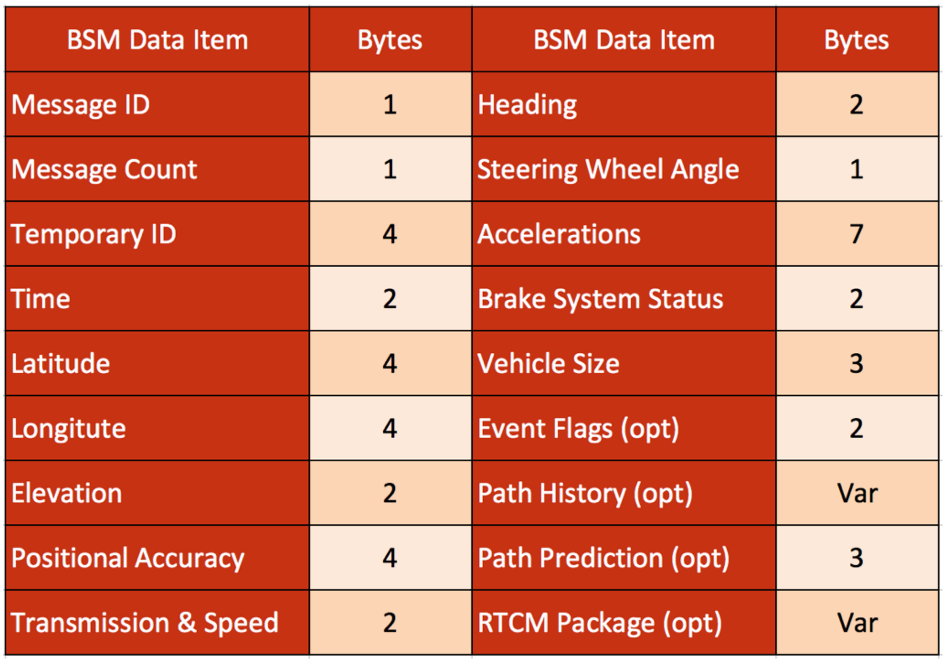
1.2.1 Overview

VANETs are developed primarily for improving traffic safety. Afterwards, more VANET applications have been designed for providing more functionalities to the vehicles and better driving experience as a part of building Intelligent Transportation Systems (ITS). These applications include traffic congestion avoidance [GGR15b], platooning by cooperative adaptive cruise control [SJB14], post accident notifications, road hazard notifications, peer-to-peer media sharing [ZZC09], e-commerce [LPA06], electronic toll collection, detection of available parking, fuel consumption optimization, etc. These VANET-based applications provide cars with much better traffic awareness, and reduce traffic congestion and fuel consumption by increasing the highway throughput (congestion avoidance, platooning, electronic toll collection, etc.). Applications like media sharing and e-commerce improve driving experience; for example, someone who cannot make it to a Lakers game due to a traffic jam can sell the tickets to a vehicle already close to the Staples Center.

1.2.2 Communication

VANET communication specifications are defined by the standards of WAVE/IEEE 802.11p/DSRC [Wir10] [SAE09] and explained in [Ken11]. Wireless channels are split into two categories: the control channel for safety applications, and service channels for non-safety applications. Cars broadcast Basic Safety Messages (BSMs) using the control channel with a frequency of 10 messages/sec. Each vehicle within the 300-meter radio range defined by IEEE 802.11p [Wir10] will receive the BSM, process and flood it to its neighbors. Figure 1.1 shows the BSM content with its fields and the corresponding size of each field.

The BSM broadcasts among vehicles will help develop a collective awareness so that preemptive actions can be taken to avoid traffic accidents. Vehicles can be aware of the statuses of other vehicles in a radius of many miles in a few seconds and avoid getting into unsafe situations. For example, if there is a car that broke down and stopped in the middle of the road after a very long and sharp curve, approaching vehicles will not be



BSM Data Item	Bytes	BSM Data Item	Bytes
Message ID	1	Heading	2
Message Count	1	Steering Wheel Angle	1
Temporary ID	4	Accelerations	7
Time	2	Brake System Status	2
Latitude	4	Vehicle Size	3
Longitute	4	Event Flags (opt)	2
Elevation	2	Path History (opt)	Var
Positional Accuracy	4	Path Prediction (opt)	3
Transmission & Speed	2	RTCM Package (opt)	Var

Figure 1.1: The information inside a BSM that vehicles broadcast

able to see the car before it is too late. However, in VANETs, since the broken-down car will broadcast BSMs, all the vehicles in the 300-meter range and beyond (due to flooding of BSMs) will hear the messages and maneuver away much before the visual contact.

In VANETs, cars can make use of three different modes of communication depending on the purpose of the currently running VANET applications. Figure 1.2 shows the use cases of all these three communication modes: Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I) and Vehicle-to-Pedestrian (V2P). Vehicle-to-Everything (V2X) is used to refer to all of these three modes at the same time. Vehicles in VANETs are also expected to have Internet access through their cellular connections (4G/LTE) and/or V2I communication. VANET communication (V2V, V2I, V2P) supplemented by Internet access aims to provide more capabilities to vehicles and more functionalities to VANET applications (informing authorities in case of an emergency, road map downloads, etc.).

Vehicle-to-Vehicle (V2V) Communication V2V communication is mainly used by vehicles to broadcast their BSMs. However, there are also many other safety and non-safety VANET applications that utilize V2V communication. Figure 1.2 shows three different use cases of V2V communication, represented by red arrows. The red and purple car are exchanging BSMs to cooperatively slow down before arriving at the intersection.

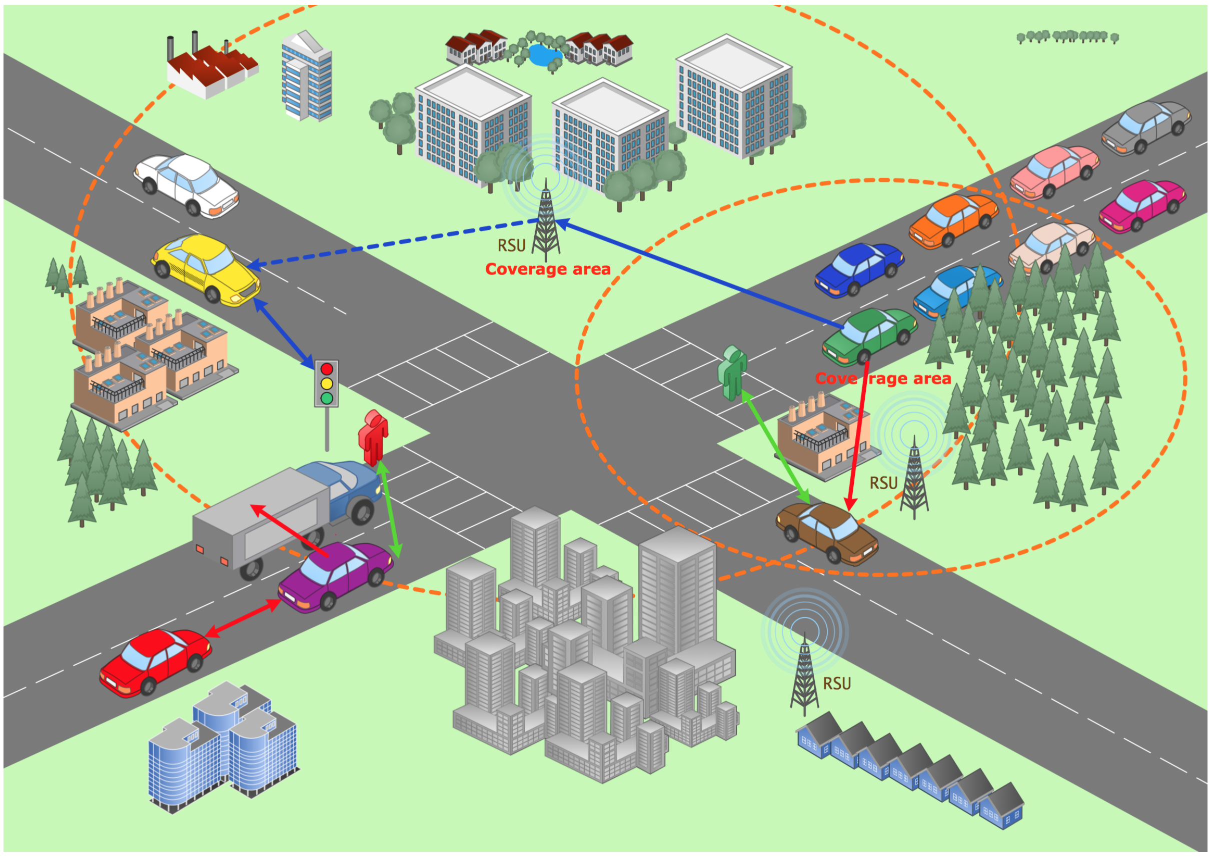


Figure 1.2: A VANET scenario where cars perform V2V, V2I and V2P communication

The purple car is currently sending a Blind Spot Warning message to the truck so that the truck would not change lanes since it does not see the purple car on the passenger side mirror. The green car is warning the brown car about the congested road it is currently on so that the brown car can avoid the right turn if possible.

Vehicle-to-Infrastructure (V2I) Communication In VANETs, infrastructure components (e.g., traffic lights, tollgates, roadside units, etc.) will also have communication capabilities. Roadside Units (RSUs) are fixed base stations that are installed on the side of road sections to serve as wireless LAN access points to the vehicles in their radio range. Since their radio range is much larger than an individual car's radio range and all RSUs are interconnected, they are mainly used to improve the overall coverage of VANETs, especially in case of a sparse vehicular network. Vehicles' broadcast BSMs will be flooded by RSUs too, thus, they will be heard by the cars in a much larger area. RSUs can also be used to alert authorities in case of accidents and sometimes for Internet access. Figure 1.2 shows two use cases of V2I communication, with a traffic light and an RSU.

V2I communication is represented by blue arrows. The green car is advertising the high congestion levels that it is currently experiencing. It is able to warn vehicles in its radio range through V2V communication; however, it has to warn the yellow car through V2I communication taking advantage of the RSU's larger radio range. In the other use case, the yellow car is approaching the intersection and would like to know whether it can make it before the light turns green for the intersecting traffic. Therefore, it uses V2I communication to learn the remaining seconds from the traffic light to be able to make an accurate decision on how to adjust its speed.

Vehicle-to-Pedestrian (V2P) Communication Because of their built-in DSRC capabilities, vehicles and infrastructure are able to participate in VANET communication. Pedestrians are not part of VANETs according to the original VANET design. However, their safety is as important as the safety of drivers. Therefore, researchers worked on the ways of making pedestrians members of VANETs. Honda designed a V2P communication protocol in order to prevent vehicles from hitting pedestrians by using DSRC-enabled smartphones [Cun13]. These smartphones will warn the vehicles in the vicinity about their owners' presence when they are crossing the road. Honda's implementation of V2P communication is explained in detail in [SYB15]. Figure 1.2 shows two use cases of V2P communication, which is represented by green arrows. Because of the truck and the house, both the purple and brown car cannot see the pedestrians currently crossing the road that they intend to use. The V2P notifications from these pedestrians cause those vehicles to stop, preventing a possible lethal situation from happening. However, some pedestrians might not have the capability of participating in V2P communication; therefore, vehicles have to be more cautious in the absence of V2P notifications from pedestrians.

1.3 Autonomous Driving

In the future, autonomous vehicles will be a large part of our lives. These cars could save 30,000 lives and prevent 2.2 million car accident injuries each year just in the US alone [Tan14]. We envision a 2020 scenario where all cars will have a certain degree of autonomy, say sufficient to allow the on-board computer to maneuver the car when its situation is "normal" and the driver wants to attend other tasks like reading a map or checking out the video of the next tourist attraction. This 2020 prediction has been widely advertised by reputable sources [Und14]. Major car manufacturers such as Audi, BMW, Ford, GM, Lexus, Mercedes-Benz, Nissan, and Volvo have already started their preparations to get a competitive edge in the autonomous car market [Tan14]. VANETs will be integrated into these cars to complement their on-board sensing and collision avoidance tools. VANETs will not only increase the collision avoidance capabilities of autonomous vehicles but also make their deployment easier and faster, since manufacturing these cars will be cheaper without the dependence on highly expensive sensors.

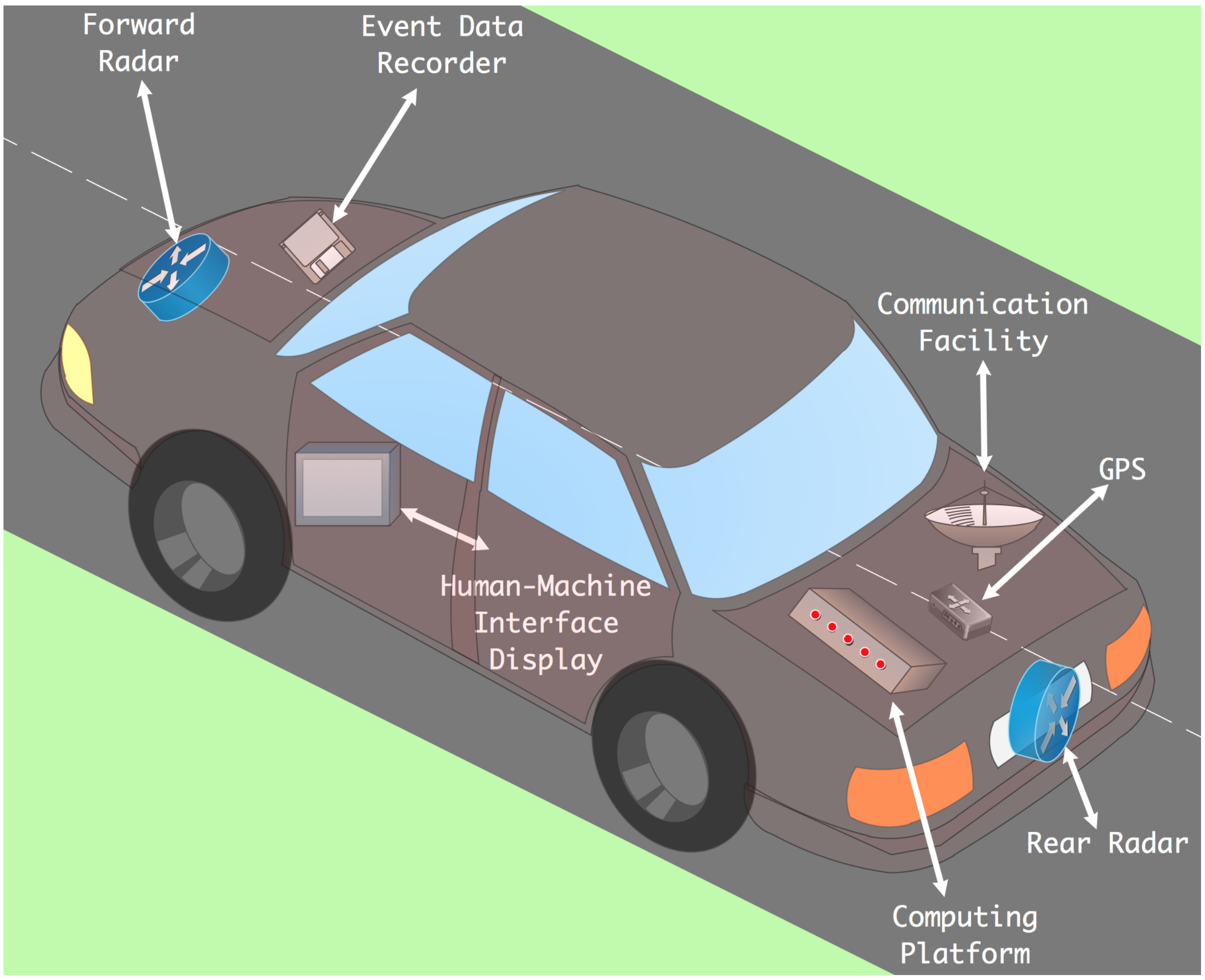


Figure 1.3: Provisional hardware components of a VANET-enabled autonomous vehicle

Figure 1.3 shows what a VANET-enabled autonomous vehicle is expected to look like.

1.4 Possibility of Vehicular Botnets

History has shown that as computing and communication capabilities are added to new environments, inevitably some fraction of the new equipment is compromised by attackers. Adding computational capabilities to automobiles and connecting them by VANETs should be expected to follow the same path. Already researchers have demonstrated that computers and internal networks within cars can be compromised and exploited, allowing varying degrees of control of the vehicles [CMK11] [IMT10] [Tut14]. In some cases, the vehicles can be compromised remotely, using V2V unlicensed network connections from a distance. Vehicles can also be attacked from incautious Internet downloading via an otherwise secure LTE. As we increase cars' cyber capabilities, the attack surface they present and the increased number of targets will certainly attract cyber attackers, and some attacks will succeed in compromising the vehicles. As the internal computers take greater and greater control over vehicular capabilities, leading eventually to autonomous cars as described in Section 1.3, the reward that a successful attack can harvest will only make compromising them a more attractive target. Existing research has already proven that compromising these autonomous cars is possible [Smi13] [Ros13]. A compromised vehicle of this kind not only offers computation and communication capabilities, but the ability to perform significant real world actions that could have severe consequences.

Car manufacturers are very likely to take some efforts to improve the security of these vehicles as both their popularity and vulnerability increase. However, based on experience with all types of systems, these efforts will not be enough to prevent vehicles from being compromised. Therefore, any research on VANET security and privacy has to be prepared for the existence of these compromised vehicles. Some attention has already been paid to the security of VANET communication [PP05], particular to ensuring that vehicles can provide strong authentication for their messages [RH07], which is important for the non-repudiation and accountability requirements for VANETs. While proper authentication is a necessary step to providing many security solutions, such solutions proposed by existing research have been primarily oriented towards preventing vehicles from cheating and lying to other vehicles and roadside infrastructure. Such research offers some leverage against attackers who compromise a single vehicle. However, history has also shown that attackers who realize that they can compromise a single machine quickly move on to compromising multiple machines, and to organizing their set of compromised machines into a distributed resource of greater total power than its individual parts.

We should thus expect that the future will bring us *vehicular botnets*, collections of autonomous or largely computer-controlled and networked vehicles under the coordinated control of remote attackers. This future brings many important questions to mind that have not yet been considered, let alone answered. What would an attacker do with such a vehicular botnet? How would he use his compromised assets to achieve his goals? How could we determine that a vehicular botnet is at work? How could we identify its members and what actions could we take to negate the damage they can do? Can we defend against such malicious vehicular botnets without identifying and "disinfecting" the vehicles? Our work is the first step in the literature towards answering these questions. Since researchers already showed that autonomous vehicles can be compromised, we will not do further research on how to compromise them but rather assume that we already have a set of these compromised vehicles when demonstrating our vehicular botnets.

1.5 Evaluation Method

The evaluation of our research was done primarily in simulation. We used the Veins simulator [Som15] (which combines the SUMO and OMNeT simulators) to conduct VANET experiments. SUMO is an open source traffic simulator for large road networks [KEB12]. It simulates vehicular traffic with realistic mobility patterns, and vehicles' interactions with the infrastructure components. OMNeT is an open source network simulator [Ope15], which supports all of the VANET communication modes that are defined by the IEEE 802.11p standard [Wir10].

The scope of our research does not permit us to purchase actual autonomous vehicles for testing, and also demonstrating our work with real autonomous cars in the real traffic is impractical and dangerous. The UCLA Campus Vehicular Testbed [CFG10] could be used for future work but further investigation is needed to assess the dangers of experimenting with vehicular botnets on this testbed.

1.6 In Closing

This chapter served as a brief overview of VANET (its description, purpose, and usage), autonomous vehicles that will soon be a part of our lives, and the possibility of vehicular botnets. We showed that VANETs could be used to improve traffic safety and for other types of applications. We argued that autonomous vehicles could be compromised and vehicular botnets could be formed with a set of these compromised vehicles. Finally, we described the simulators that were used for the evaluation of our work. In the subsequent chapter, we discuss how existing research cannot cope with our vehicular botnets, and describe our design for organizing such botnets.

CHAPTER 2

Vehicular Botnets

2.1 Introduction

Vehicular botnets open up new possibilities for powerful and disastrous attacks that have never been considered before, as a result of the greater total power of the compromised vehicles—*vehicular bots*—than an individual malicious car. The owners of these vehicles are not aware that their cars are compromised while, secretly, they perform attacks under the coordinated control of a remote attacker—*botmaster*—who controls the botnet. While vehicular botnets render the existing VANET security solutions ineffective due to their nature, as discussed in Section 2.2, they also provide botmasters with a significant level of anonymity since the vehicular bots perform attacks on their behalf. In order to harvest the power of vehicular bots, however, a communication protocol needs to be designed, through which they can coordinate their attacks with minimal interaction with the botmaster and without getting detected—a challenging task.

In this chapter, we discuss the existing research on VANET security and why the proposed solutions cannot cope with our vehicular botnets. We then present the design details of GHOST [GRG16], our vehicular botnet communication protocol that hides itself in the VANET control channel traffic. Designing such a protocol and analyzing its strengths and limitations are imperative to understanding the inherent characteristics of vehicular botnets and identifying ways to defend against them.

2.2 VANET Security

The existing work on security of autonomous vehicles and VANETs is primarily related to bad behavior or compromises of individual vehicles, with a related body of research on privacy issues in VANETs. [RH07] provides a good survey of the early research on these topics. This paper mentions the possibility of multiple cooperating malicious entities in the VANET, but does not discuss them in detail. A significant body of work on defenses against security problems in VANETs focuses on ensuring the correctness of cryptographic authentication in VANETs [HCL04]. The VANET authentication mechanism involves using Public Key Infrastructure (PKI), and each vehicle signing its messages with a private key issued to it by authorities. While this prevents attackers from spoofing other cars' identities, it does not defend against our vehicular botnets since vehicular bots do not perform identity spoofing. In addition, this mechanism may ensure that all messages can be properly attributed to their sender, but experience with other forms of botnets shows that merely knowing who participated in attacks is at best a first step toward solving the problem.

[KSM12] focuses on categorizing vehicular crimes in a cyber-environment. It is crucial that lying and misbehaving nodes are caught, otherwise critical system and safety components can be tricked. The research community has acknowledged this problem. A lot of work has gone into the detection of position faking, Sybil nodes and fake (i.e., ghost) vehicles. More recent work has looked at tamperproof hardware in cars [PP05] [GB08] [Evi08], and analysis of messages sent by VANET members to evaluate plausibility [GGS04] [EMT02] [BSB10] [BMB12] [SLS08] [LSK06] [YCW07] [KSD10] [SFH11] [BNP12] or trustworthiness of the sender [MV13] [DFM05]. [PFK11] considers cooperating attackers, but does not discuss them in the context of a botnet.

While the general approach of analyzing messages for plausibility and evaluating trust in nodes has promise, it does not attempt to identify cooperating attackers, or at best assumes that a majority of the data comes from honest vehicles. Vehicular botnets, however, can be deployed such that they create a significant majority in a certain vicinity, or located strategically so that they poison enough cars to allow a successful execution of a botnet-originating attack. Therefore, the existing security solutions cannot defend against our vehicular botnets.

2.3 GHOST

Vehicular bots need to utilize a vehicular botnet communication protocol for the collaboration crucial in their attacks. Therefore, investigating the characteristics of such communication is important for defending against vehicular botnets, especially considering that the most effective way to stop a botnet is taking down its communication [NAP13]. This work is the first that addresses vehicular botnet communications.

The functional requirements of a vehicular botnet communication mechanism are similar to those of Internet botnets; they need to exchange sufficient information to allow attacks to be performed, while protecting the identity of the botmaster and bots, avoiding detection, and being resilient to the loss of several bots. However, how these requirements can be satisfied by vehicular botnets is fundamentally different than Internet botnets, because the characteristics of communication over the Internet and VANETs are fundamentally different. Unlike the Internet, in VANETs vehicles have to use a shared medium to communicate, which every other vehicle can listen to (i.e., control and service channels). In order to protect the identity of the botmaster, the same approach with Internet botnets can also be used by the vehicular botnets, which is using multiple levels in their command and control (C&C) hierarchy to reach the botmaster. However, preventing detection during the bot communication is much more challenging in VANETs than it is in the Internet. In this work, we present our vehicular botnet communication protocol, GHOST, that exploits the existing vulnerabilities in the VANET standards to remain hidden. It satisfies these botnet communication requirements, while at the same time being restricted by the characteristics of VANETs. We verify via simulation that GHOST is indeed infeasible to detect unless the current VANET standards are improved.

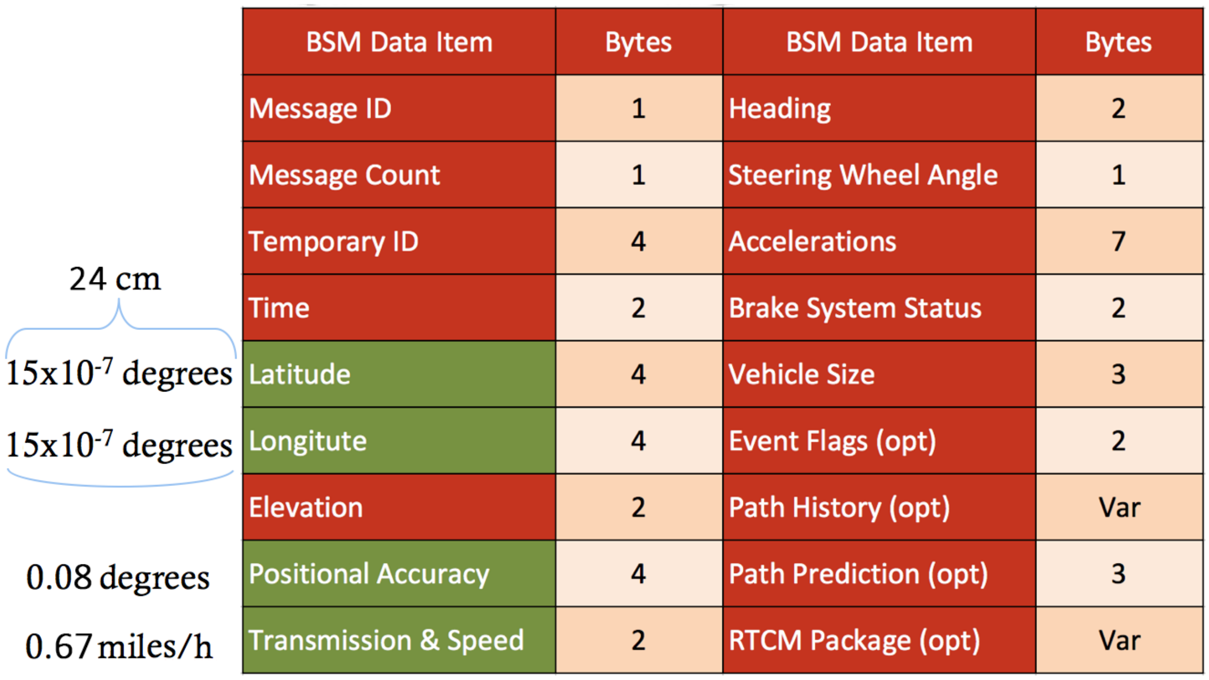
2.3.1 Vehicular Botnet Communication

Creating a vehicular botnet requires building a communication infrastructure to organize and control the compromised vehicles so that the attacks can be performed cooperatively by these vehicles. Since using vehicles' Internet connections for communication among vehicular bots will raise a flag, our vehicular botnet communication needs to use the existing VANET infrastructure. Using existing communication mechanisms of the current Internet botnets for vehicular botnets is not feasible since the Internet architecture is fundamentally different than the VANET architecture. Therefore, we designed a vehicular botnet communication protocol that takes advantage of the existing VANET infrastructure to control the vehicular bots and facilitate collaboration for vehicular botnet attacks.

2.3.1.1 Design Overview

The VANET communication infrastructure has two different wireless channels that vehicles can use: the control and service channels. BSMs are broadcast through the control channel; other non-safety applications use the service channel. Using the service channel to exchange many botnet messages among a particular set of vehicles would be suspicious, so we decided to hide the vehicular botnet communication in plain sight. Vehicular bots will inject their messages into BSMs, which are already being flooded across the network, but no vehicle except our vehicular bots will know the existence of these secret messages and be able to decode them. We designed the injection mechanism in a way that the altered BSMs will not look suspiciously different than the normal BSMs. In addition, we do not inject botnet data into every BSM and we constantly change the injection frequency to further decrease the possibility of raising suspicion; details are explained in the subsequent sections.

Not all the fields in a BSM can be altered without causing implausible data and raising any flags. We alter only four fields in a BSM where an injection can stay undetected based on the resulting value of each field. Figure 2.1 shows the content of a BSM according to the IEEE 802.11p standard [Wir10]. The fields that we use for injection are colored green: "Transmission & Speed" for the vehicle's current speed, "Positional Accuracy" for the noise in calculation of the vehicle's heading relative to true north, "Latitude" and "Longitude" for the vehicle's GPS location. We inject half a byte to each field by replacing the least significant 4 bits of the field. We limit the size of each injection to half a byte so that the changes due to the injection in the field values are not suspicious. As shown in Figure 2.1, our injection can change the value of the speed field at most 0.67 miles/hour, positional accuracy field at most 0.08 degrees and GPS position at most 24 centimeters (on the equator)—completely natural variations even without an injection.


24 cm
$15x10^{-7}$ degrees
$15x10^{-7}$ degrees
0.08 degrees
0.67 miles/h

BSM Data Item	Bytes	BSM Data Item	Bytes
Message ID	1	Heading	2
Message Count	1	Steering Wheel Angle	1
Temporary ID	4	Accelerations	7
Time	2	Brake System Status	2
Latitude	4	Vehicle Size	3
Longitute	4	Event Flags (opt)	2
Elevation	2	Path History (opt)	Var
Positional Accuracy	4	Path Prediction (opt)	3
Transmission & Speed	2	RTCM Package (opt)	Var

Figure 2.1: BSM fields where botnet messages will be injected into and the effect of the injection on the field values

2.3.1.2 Vehicular Botnet Message

The size of each vehicular botnet message is 2 bytes, which will be divided into four equal parts to be injected into a BSM. Each botnet message has 3 fields as shown in Figure 2.2:

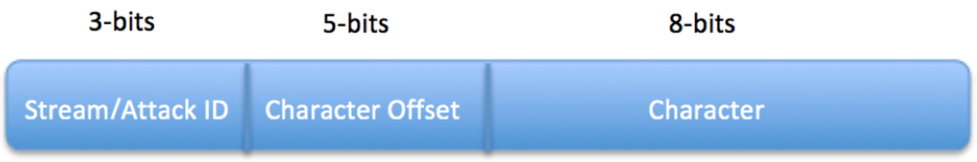


Figure 2.2: Botnet message that will be injected into BSM

Stream/Attack ID: In a vehicular botnet, the same botnet communication infrastructure has to be used for different attacks, sometimes even simultaneously. Each bot should be able to separate data streams from each other based on which attack each stream belongs to. Therefore, we have a stream/attack ID field in the botnet message which specifies which stream this message belongs to so that bots can separate different data streams. We allocated 3 bits in the botnet message for the stream ID, which supports performing up to 8 different vehicular botnet attacks simultaneously.

***Character Offset*:** Due to the limited size of injection in order to stay under the radar, each botnet message has only a single byte allocated for the payload. The data that a bot needs to send is divided into individual characters, each of which is sent with a separate botnet message. Since BSMs altered by these botnet messages might arrive out of order at the receiver bots, we allocated 5 bits in the botnet message to hold the character offset so that the data received can be reconstructed with the original order. Therefore, at most 32 characters can be sent in a single stream due to 5-bit character offset; multiple streams can be used if larger data needs to be sent, decreasing the number of simultaneous attacks possible.

2.3.1.3 Communication Secrecy

Our vehicular bots do not communicate with each other over the Internet, a separate wireless frequency or an individual service channel in the VANET, existence of which might get detected. Our botnet communication is completely hidden in BSMs, which are broadcast and flooded to the entire VANET. Detecting the existence of secret messages embedded in BSMs is difficult since the changes in the field values after injections are negligible and expected due to the noise in sensor readings. Our mechanism discussed in the subsequent sections is designed to provide confidentiality if anyone ever suspects that vehicular botnets use BSM broadcasts to communicate.

Even though one cannot guess which and how many fields bots are injecting their messages into and the size of each injected botnet message by just monitoring BSM exchanges, we still decided to assume that this information can be obtained just in case police might apprehend one of our bots and access the source code of our vehicular botnet software.

In Section 2.3.1.6, we show that, even in this situation, our mechanism can still ensure the confidentiality of the exchanged botnet messages by the use of tokens and periodic password updates. In the next few sections, we describe how a compromised vehicle joins the vehicular botnet and the mechanism by which it periodically mutates the injection scheme.

2.3.1.4 Initialization of Bots

Bots do not use their Internet connection to communicate with the other bots. However, they will use it for occasional communications with the botmaster. There are two packet types for this communication between the botmaster and bots: a *password request* packet and a *password response* packet. These packets are used by the botnet not only for the initial setup, but also for synchronization and confidentiality purposes even after the initial setup, which is further explained in Section 2.3.1.6.

When a vehicle first gets compromised, our botnet software is downloaded onto the vehicle. The software comes with a unique token that is different for each compromised vehicle; these unique tokens are generated by the botmaster and can only be used once and only for the initial setup. This way, it becomes infeasible for authorities or any other party to join the vehicular botnet and tap the botnet communication, details of which are explained in Section 2.3.1.6. The compromised vehicle will send a *password request* to the botmaster for the first time with this token and its "Temporary ID" (pseudonym) in order to obtain a unique bot ID and the current password in the botnet, and register its pseudonym. Pseudonyms are unique abstract identifiers embedded in certificates and BSMs of individual vehicles, as required by the IEEE 802.11p standard [Wir10] (see Figure 2.1). After receiving this *password request*, the botmaster will then check its *unused token database* and confirm if the token exists. If it exists, the botmaster will generate a unique bot ID, add the (bot ID, token) pair to its *used token database*, add the (bot ID, pseudonym) pair to its *bot database*, and send a *password response* that contains this bot ID, the current password in the botnet and a list of all the pseudonyms in the *bot database* back to the compromised vehicle. Then, the vehicle will overwrite its *bot list* with this list sent by the botmaster.

2.3.1.5 Botnet Message Broadcast

The current password in the botnet is used to configure the injection mechanism. It is constantly updated by each bot at the same time, thus, the injection configuration mutates with it. Since the next password is generated from the previous password and every bot updates its password at the same time in a distributed manner, all the bots will be using the same injection mechanism at all times despite the constant mutation. Bots synchronize their times based on the GPS timestamp, avoiding issues of clock synchronization. Autonomous vehicles are certain to have good GPS equipment, given that even many car models on the market today have one; hence, this GPS timestamp will be readily available. The *password structure* in each bot is designed as follows:

***Password*:** The current password in the botnet. All the following values are set based on the current value of the password.
***Injection order*:** This determines which permutation of the half bytes of botnet message will be injected to the four fields (e.g., fourth half byte is injected to the longitude field while first is injected to the speed field in the BSM).
***Encoder/Decoder ID*:** This specifies which encoder-decoder function pair should be used by both sender and receiver bots to encode the character sent and decode the character received.
***Injection Frequency*:** This determines the frequency at which injections on BSMs will occur (e.g., inject every fifth BSM).
***Next Password Update Time*:** This specifies the time that current password must be updated by each bot so that all bots can switch to the same configurations without any coordination.

Each bot continuously checks BSMs, at a frequency determined by current configuration, to extract any injected character. First, each bot will check if the pseudonym associated with the BSM is in its *bot list*, which is the only way to understand whether the BSM is coming from a bot or not. If it exists in the list, then each bot will extract the 2-byte botnet message according to the current active injection configurations, and check whether it is a valid message or not. An invalid botnet message is indicated by setting all its bits to 1. The reason that this special botnet message exists is that every bot has to inject something to the BSM when it is time to inject, regardless of whether there is a character to send in the bot's message buffer. Therefore, each bot needs to inject this special botnet message (all bits are set to 1) if all of its message buffers are empty, to let other bots know that it did not have any character to send. If the extracted botnet message is valid, it will be placed at the correct offset in each receiver bot's message buffer belonging to the attack specified by the *Stream/Attack ID* in the botnet message.

If not, the extracted botnet message will be discarded.

When botmaster wants to send a message to all of the bots, it chooses one or more representative bots based on the sparsity and number of the bots. It sends the message to the representative bot(s) over the Internet, along with the associated *Stream/Attack ID*. The representative(s) will break the message into individual botnet messages with the corresponding character offsets and attack ID, and inject one botnet message at a time to BSMs according to the current active injection configurations. Received botnet messages will be saved by the other bots in their associated message buffers. When the next injection time comes, the other bots will then randomly choose a character among all of the received characters so far in their message buffers, and broadcast it like the representative(s) according to the configurations at that time.

2.3.1.6 Discussion on Secrecy and Consistency

Our botnet communication is resilient to attempts by the authorities to eavesdrop on the messages being exchanged among bots. Without knowing the current password, no one will be able to guess the current active injection configurations and reconstruct the botnet message. Before one might figure out the current configurations by brute force, they will already have changed due to the constant password updates. Successful brute force in a timely manner is infeasible because one needs to consider every BSM as injected, try all x! * y combinations (x=number of fields, y=number of encoding/decoding functions) and filter out the combinations that result in invalid messages before the password is updated.

One can try to brute force a token to be able to tap the botnet communication. However, each token is created in the moment of compromising a vehicle and immediately used by the compromised vehicle to join the vehicular botnet. Therefore, window of opportunity to brute force this token is extremely small, making this approach infeasible.

The only problem that might occur is that police might apprehend one of our bots and access the botnet communication software. Police can obtain the token associated with this captured bot, but due to not allowing a token to be used more than once, the token cannot be used by the police car to join the botnet. However, police can still use the captured bot itself to eavesdrop on the communication; therefore, we implemented a technique to remove the apprehended bots from our vehicular botnets. The botmaster generates a new random password every 13 seconds, which is independent from the previous password. It sends a *password response*, which contains this password and the list of current bots' pseudonyms, to all the bots in its *bot database*. When the botmaster detects that one of the bots got apprehended, it will remove the bot from its *bot database* and its associated token from the *used token database*. Therefore, the password generated by the botmaster will not be sent to the apprehended bot, and the bot list that is sent to all the other bots will not contain the captured bot. This way, the apprehended bot will be out of sync with the other bots and will not be able to extract botnet messages properly. As a result, even if police obtains the current password in the botnet communication, it will be obsolete in at most 13 seconds after the apprehension is detected. How a botmaster can detect the apprehension of bots is for future work. One way would be making the bot send a message to the botmaster if the executable of its botnet communication software is accessed for reading. Read—rather than execute—access will indicate an attempt for reverse-engineering.

Between these random password generations, the botmaster keeps updating its password based on the previous password periodically like all the other bots do. Bot vehicles, which were turned off by their owners for some time because they were parked, will miss several password updates and not be able to communicate with the other bots. Therefore, when they get turned back on, they will send a *password request* to the botmaster with their valid bot IDs to receive the current active password that the other bots are using in order to catch up with the current botnet communication.

As aforementioned, bots use their *bot list* and the pseudonyms associated with received BSMs to determine if they are sent by the bots. However, as mandated by the IEEE 802.11p standard [Wir10], every vehicle updates its pseudonym periodically for privacy reasons by using one of the mechanisms discussed in Section 3.3.1. Therefore, every time a bot updates its pseudonym, it will send a *password request* to the botmaster with its bot ID and new pseudonym. The botmaster will then update its *bot database* so that it can be shared with the other bots through the periodic *password response*.

2.3.2 Evaluation

	Average	Max	Theoretical Max
Latitude (d)	5.02E-07	7.67E-07	15E-07
Longitude (d)	5.25E-07	7.73E-07	15E-07
GPS Position (cm)	8.08	12.11	24
Positional Accuracy (d)	0.02	0.03	0.08
Speed (miles/hour)	0.29	0.40	0.67

Figure 2.3: Changes in the values of the injected fields during the experiment

First, we show that it is infeasible to detect our botnet communication based on its use of BSMs. Figure 2.3 shows the average changes on the four BSM fields due to the injection of botnet messages, as well as the maximum changes observed during our experiments. It can be seen that actual changes on these fields are much lower than the theoretical maximum values that we anticipated in Section 2.3.1.1. Changes in the speed values stay under the speedometer errors tolerable by the UN regulations [UN 16]. Changes in the GPS position and positional accuracy are much lower than the expected errors in the sensor readings ($\approx$ 1 meter for the state-of-the-art GPS sensors [US08] and 1 degree for the magnetic compasses [Mil14]). These results show that it is infeasible to detect our botnet communication.

Packet Loss (%)	0.0	0.5	1.0	1.5	2.0
Average Sync Time (s)	20.43	20.84	18.02	20.91	22.80

Figure 2.4: Average synchronization times with different packet loss rates

Our botnet communication is resilient to losses of BSMs. We tested our systems with different packet loss rates and measured, for each rate, the average time it takes for a bot to receive the complete message associated with an attack (average sync time). Figure 2.4 shows average sync times mostly increase as the loss rate increases, but the changes are negligible; bots receive complete message in a timely manner.

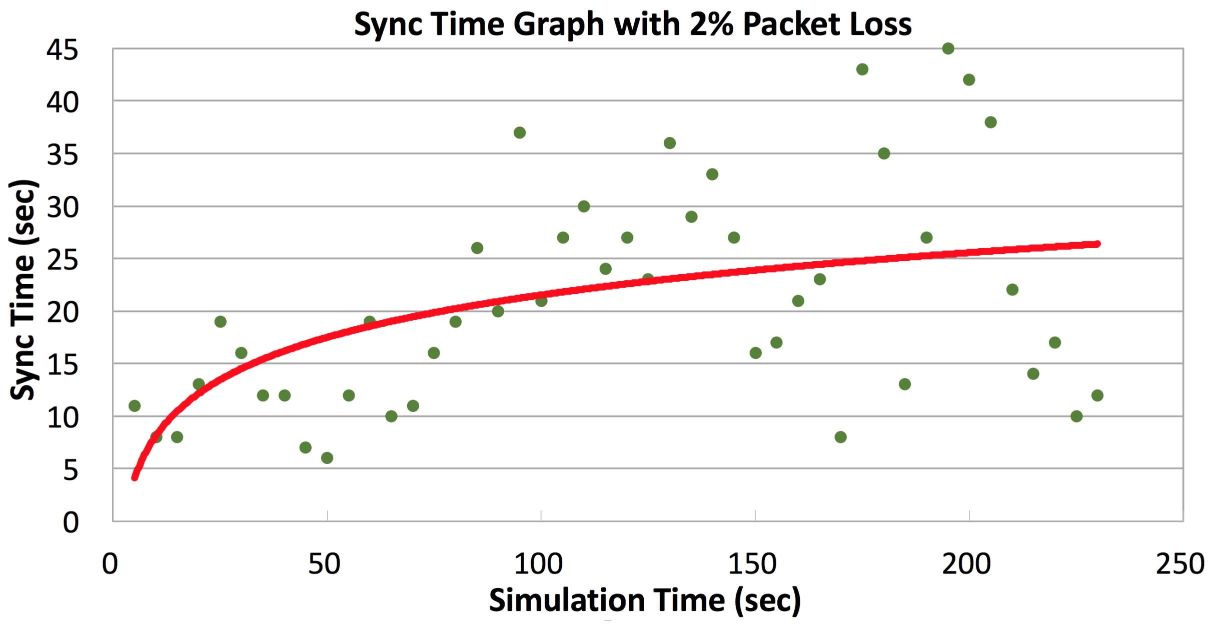


Figure 2.5: Sync time graph when the packet loss rate is 2%

Figure 2.5 shows how resilient our botnet communication is to packet loss and explains the reason of the results in Figure 2.4. Sync times increase throughout the simulation as more packets are lost, but our system quickly adapts to the lossy links and stabilizes at acceptable values. It stabilizes sync times by a randomization strategy we designed for advertising botnet messages. For each botnet message, bots broadcast a character at a random index in their message buffer so that the variety of characters being advertised at each time is maximized. As a result, the probability for a bot to receive a character which it has not received yet will increase and sync times will improve compensating the existing packet loss rate.

Bot Percentage (%)	5	10	20
Average Sync Time (s)	11.10	12.96	20.43

Figure 2.6: Average synchronization times with different botnet percentages

We also investigated how scalable our botnet communication is when a large number of bots use the botnet communication, by testing our system with different bot percentages. Figure 2.6 shows the average sync times when 5%, 10% and 20% percent of the vehicles in the simulation are bots. As expected, the average sync time increases with the bot percentage due to the computation overhead and packet loss introduced by it. However, the increase is logarithmic and therefore tapers off before exceeding acceptable sync times.

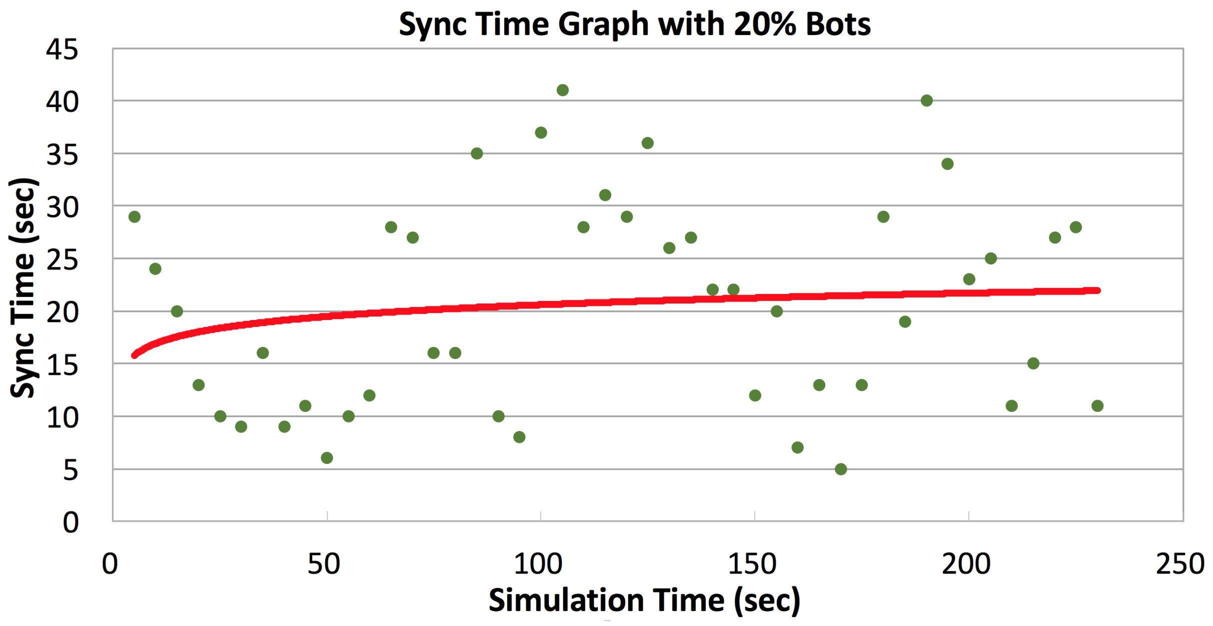


Figure 2.7: Sync time graph when 20% of the cars in the simulation are bots

Figure 2.7 shows how fast our botnet communication adapts to the overhead caused by the large number of bots when the bot percentage is 20%. It stabilizes sync times this quickly by exploiting the large number of bots for the aforementioned randomization strategy; when the number of bots is high enough, this strategy successfully compensates for the packet loss induced by the overhead. Also, the stabilization of the sync times in Figure 2.7 is much faster than it is in Figure 2.5 with high packet loss, since the effectiveness of our randomization strategy increases as the number of bots gets higher.

2.4 In Closing

This chapter provided a survey of existing work in VANET security and argued that the current defense mechanisms cannot mitigate our vehicular botnets. We asserted that the most effective defense against vehicular botnets, as all other types of botnets, is investigating the characteristics of the communication they use to perform their attacks and designing defense mechanisms to disrupt it. As a step toward such defense, we designed the first vehicular botnet communication protocol in the literature to allow us to study it. This way, we can design a general defense that can prevent all types of vehicular botnet attacks since they all will have to use a botnet communication protocol.

CHAPTER 3

Vehicular Botnet Attacks

3.1 Introduction

Vehicular botnets enable powerful and dangerous attacks that neither single malicious vehicles nor Internet botnets could perform. They widen the attack surface of VANETs with novel attacks due to the collective power of vehicular bots, and these attacks could have physical and possibly fatal consequences—unlike Internet botnets. In addition to investigating vehicular botnet communication for an all-inclusive defense against vehicular botnets, it is also crucial to understand the attack surface introduced by them and determine possible countermeasures against their attacks as a supplement to this defense. Therefore, in this chapter, we demonstrate three powerful attacks performed by our vehicular botnets. Any coordination needed among vehicular bots for these attacks is performed through GHOST. Each attack represents a different category of security threats posed by vehicular botnets so that we can conduct a comprehensive study on a wider variety of vehicular botnet attack characteristics. These categories are having physical consequences, violating privacy, and threatening other types of systems and networks than VANETs.

We first present our congestion attack [GGR15a], which can cause traffic congestion on any road of interest, making it and other roads surrounding it virtually unusable. This attack shows that vehicular botnet attacks can have physical consequences. We then demonstrate BOTVEILLANCE [GRG18]—our surveillance attack—which violates one of the most fundamental requirements for VANETs, location privacy. We also describe its novel built-in localization mechanism, INTERLOC [GKR17]. BOTVEILLANCE shows that vehicular botnet attacks can violate the privacy of VANETs in addition to having physical consequences. Finally, we present RIoT, a rapid exploit delivery mechanism utilizing the mobility and collective communication range of vehicular bots to compromise—along their paths—as many Internet of Things (IoT) devices as possible in any area of interest. RIoT shows that vehicular botnets are threats not only to VANETs; they can also be used by attackers as an effective tool to perform powerful attacks against other important systems and networks. The possible countermeasures against individual vehicular botnet attacks are discussed in Chapter 5.

3.2 Road Congestion Attack

Road congestion is a serious problem for big cities with high populations [Sch08]. Solutions like Google Maps traffic data, which report congestion in broad areas, are often inadequate for micro-traffic management ("what is going on around the corner that has caused all traffic to stop?"). For such applications, Google Maps and WAZE reports are generally obsolete as their web refresh rates (for scalability reasons) are slower than the congestion change rate. Cars that are already in the area can give the most up-to-date information about congestion—on the order of minutes rather than tens of minutes. Using VANETs, and more precisely V2V VANET protocols, such timely information exchange can be achieved; cars can then choose their routes around a sudden traffic jam, using the information obtained from other cars. Since V2V congestion avoidance is a commonly agreed benefit of VANETs [YB09], it is a reasonable target for an attack.

In this work, we present a damaging vehicular botnet attack targeting this V2V propagation of traffic congestion information. Our attack can cause heavy congestion on any chosen road by exploiting the common vulnerabilities of existing VANET congestion avoidance mechanisms. It can increase the trip times of vehicles that use the targeted road by orders of magnitude. Furthermore, the effect of our attack does not stay confined to the targeted road; the congestion quickly spreads to the neighboring roads and significantly affects the entire urban grid.

The main purpose of this work is to demonstrate that vehicular botnet attacks can have physical consequences, unlike Internet botnets. As attackers often use Internet botnets to perform DDoS attacks to congest Internet links, our attack causes road segments to be flooded with *physical* traffic, making them almost unusable. The physical consequences of the congestion attack can be used for economic and political [Huf14] incentives. For example, our attack can be used in scenarios as serious as delaying police arrival to a robbery scene, blocking emergency vehicle access to an area targeted by a terrorist attack, or as simple as increasing congestion around a specific store to favor its competitor. In serious scenarios like the ones described above, the consequences of our attack can be devastating—perhaps fatal.

3.2.1 Related Work

In addition to enhancing traffic safety, deployment of VANETs and V2X communications can certainly aid traffic congestion in urban environments. [YB09] shows via simulation that vehicular communications and dynamic route planning techniques can provide significant reduction in total trip time. They can also reduce the average number and density of vehicles in congested areas. [BGS10] proposes a system based on V2V communications that detects congestion, without requiring additional infrastructure or sensors. It addresses congestion detection, but does not fully consider the dissemination of congestion information or rerouting in response to congestion. [FBS10] considers the problem of dynamic route planning and how to estimate a car's remaining trip time. It verifies that latest trip times reported by other vehicles are quite useful to calculate this and congestion levels in the area. It proposes and evaluates heuristic improvements to the *trip time* metric as well. On the other hand, in [MR13], vehicles monitor average speeds to detect and quantify congestion. They can then perform aggregation of their measurements and adaptively broadcast their results.

For detecting and avoiding congestion, efficient automated protocols are needed to share congestion readings among vehicles. Earlier work in this area is discussed in Section 3.2.2, details of which are core to the research described here.

Since we do not make use of Sybil attacks or position faking in our congestion attack, our attack is resistant to the plausibility filters discussed in Section 2.2; identifying the information that our vehicular bots advertise as outliers is difficult. Nevertheless, we pay extra attention to make vehicular bots lie within the limits established by the plausibility filters' thresholds, since evicting misbehaving nodes (e.g. by revoking certificates) in a timely manner is possible as described in [RPA07].

3.2.2 Congestion Avoidance

The attack we introduce depends on exploiting vulnerabilities in proposed V2V congestion avoidance mechanisms for VANETs, so before discussing the attack, we will briefly describe these mechanisms.

VANETs can be used to reduce the traffic congestion problem by taking advantage of the communication capabilities of cars (e.g. 4G/LTE, V2V, etc.). [BGS10], [FBS10] and [MR13] discuss the ways to measure and detect congestion so that cars can avoid it by using dynamic rerouting strategies with trip time predictions. There are two approaches to design such a congestion avoidance mechanism, cloud-based and V2V approach. [HSL13] and [KG11] propose mechanisms that follow a centralized approach, using cloud-based design. Vehicles upload their trajectory information to access points that help facilitate predictions on future traffic, based on current traffic load and other vehicles' intended destinations. Access points then respond with time-efficient paths to the inquiries from the cars on the map. [CZL10] and [LMM11] are examples of V2V, totally distributed congestion avoidance approaches. These two methods offer different, complementary advantages. [LMM11] opts for periodic broadcasts of the subset of vehicles' congestion measurements. The appropriate subset is determined by the freshness of readings and their map coverage. On the other hand, [CZL10] focuses relatively more on reducing communication costs (e.g., using cache structures and queries with expiration times) while sharing an adequate amount of information.

The cloud-based scheme can achieve optimal traffic assignment on multiple, parallel routes over a large geographic area (say, the entire metro area) because of the centralization of all the requirements. The routing instructions are valid for the medium and long term. However, routing changes (around a congested area) are typically announced several minutes after the onset of congestion. The V2V approach, on the other hand, can more promptly react to local congestion problems and offer rerouting solutions in fractions of a minute. In contrast with cloud-based systems that receive congestion information from vehicles on a one-to-one basis, the V2V congestion avoidance mechanisms rely on dissemination of the congestion levels observed by the cars in order to make better navigation decisions.

All V2V schemes share the same vulnerabilities that our attack exploits: they completely trust and forward congestion information without trying to detect outliers or incorrect information. We perform our attack on the functionalities of the V2V congestion avoidance mechanisms proposed in the literature. In order to simplify our experiments (without affecting the general validity of our attack), we created a benchmark V2V congestion avoidance strategy [GGR15b] that presents some changes with respect to the schemes found in the literature. First, we use a reactive instead of proactive approach. Namely, cars issue congestion request-response packets when they need the information instead of periodically disseminating congestion information. Second, we precompute all the possible routes from every origin on the map to any destination so that our simulator does not incur the high computational cost of calculating these routes in real time. These changes are purely for optimizing experimentation; they do not introduce any vulnerability in addition to the already existing ones in all V2V schemes, as discussed in Section 3.2.5, or favor our attack in any way. In the subsequent sections, we describe the implementation, how cars generate and store congestion information, and how they choose the least congested routes.

3.2.2.1 Message Types and Information Exchange

For the communication among all cars, we assume the standard signal range for IEEE 802.11p, which is 300 meters [Wir10]. Two types of messages are implemented in our congestion avoidance simulation: congestion request and congestion response messages—separate from regularly broadcast BSMs.

Congestion Request Messages When a car approaches the end of a road and can choose from two or more candidate routes to its destination, it will need congestion measurements from other vehicles in order to make the choice that will minimize its remaining trip time. Hence, it broadcasts a congestion request message to all nearby cars in its communication range to obtain this information. The content of this message is a list of *roads of interest*; these are the roads that form the candidate paths to the destination. The car will then choose the least congested route based on the congestion response messages it gets.

Congestion Response Messages A vehicle sends a congestion response message only when it receives a congestion request message and there are related entries in its congestion information database, ensuring that no superfluous information is transmitted. The congestion response message includes congestion information, available at the receiver, about the *roads of interest* inquired in the congestion request message.

3.2.2.2 Congestion Information Database

Creating and Storing Congestion Information In order to store and exchange congestion measurements, vehicles make use of *congestion information structures.* Each structure consists of the following fields:
***Creator ID*:** The unique ID of the vehicle that created this measurement.
***Edge ID*:** The unique ID of the one-way road that this measurement belongs to.
***Average Speed*:** The average of the speed readings from all the vehicles on the same road as the car that created this measurement, along with the car's own speed readings.
***Timestamp*:** The time of the measurement's creation to ensure the freshness of measurements and prioritize the most recent ones.

Maintaining the Congestion Information Database The congestion information database consists of the vehicle's own measurements and the measurements learned from others as a result of the congestion request-response protocol.

DB is an abbreviation for the local congestion information database of a car, which consists of zero or more congestion measurement entries denoted by m. Let E be the set of all the edges (roads) in the map: $DB = \{m_1, m_2, m_3, \dots, m_n\},\ 1 \leq n \leq |E|$

There are two cases in which a measurement gets inserted to the database. The first is when the car creates its own measurement by averaging its own speed readings, together with the readings from the other cars on the same road it is passing along—collected until the car reaches the end of the road. The second is when a measurement is obtained through a congestion response from another vehicle.

t_α is the time threshold that determines when a measurement created by the car itself becomes outdated in its congestion information database. Cars prioritize the measurements that they created themselves; they will ignore even fresher measurements heard from other cars if their own recordings are not at least t_α older than the received measurements. Note that this modification to the existing congestion avoidance mechanisms actually makes it more secure against our attack.

Least Congested Route Selection When a car approaches the end of a road, it calculates what its *roads of interest* are in order to differentiate a less congested route and sends out a congestion request asking for these roads. The congestion responses received will be merged with the congestion information database of the car. Afterwards, the routing decision becomes a set of arithmetic operations; the trip time for each of the k candidate routes will be computed, and the car will choose the one with the lowest trip time. Let $RSet$ be the set of the k candidate routes. Each $R \in RSet$ is a series of edges which are represented by E. Therefore, the selection of the least congested route from $RSet$ is performed using the following calculation:

$$R_{Chosen} = \min_{\forall R \in RSet} \Big(\sum_{\forall E \in R} Length(E)/AverageSpeed(E) \Big)$$

3.2.3 Congestion Attack Design

3.2.3.1 Types of Vehicles in the Attack

In our attack, there are three different types of vehicles: parked bot cars without any driver inside, compromised cars with drivers, and uninfected cars. Only the parked bot and compromised cars perform the attack, with different capabilities and roles as the members of our vehicular botnet.

Parked Bot Cars The attackers control a certain number of autonomous cars spread all around the map without any driver on board. These vehicular bots are parked in random places and perform malicious advertisements of bogus congestion information if they are close to the area being targeted. The parked bot cars are inactive if there are enough compromised cars to cover the targeted area in their wireless communication range; they know how many vehicular bots are in that region, along with their positions. During the attack, when the parked bot cars sense that certain portions of the map are not sufficiently covered by the compromised cars, they will coordinate with each other and will be directed to cover those areas. They will also advertise their bogus congestion measurements with “significantly fresh” timestamps. Within due time, their advertisements will dominate the congestion responses from the uninfected cars.

Compromised Cars These cars have been compromised by attackers using the methods discussed in Section 1.4, yet they have legitimate drivers on board, who have no idea that their cars have become vehicular bots. They have legitimate routes and minimize their trip times, as the uninfected cars do, using the congestion avoidance mechanism. They cannot perform any suspicious action that might alert their owners because this could cause the owners to disinfect their vehicles. However, since the owners have neither any control over nor any knowledge of what congestion information their cars create and exchange, these cars are used to perform malicious advertisements in our attack. They still broadcast congestion requests to obtain the necessary information from other vehicles to choose the least congested route for themselves; however, when they receive a congestion request, they respond with bogus information as the parked bot cars do.

Uninfected Cars These cars are the victims of our attack; they honestly follow the congestion avoidance protocol in the hope of minimizing their trip times. They share the congestion measurements in their database, which are either generated by themselves or heard from other cars, without any malicious alteration. However, considering that vehicular bots’ advertisements can overwhelm the legitimate congestion responses (both in terms of number and freshness of measurements), the uninfected cars will end up “honestly” disseminating the malicious advertisements originating from the vehicular bots. This will speed up the effect of our attack. Examples of this behavior are given in the subsequent sections, together with experimental results showing that higher percentages of vehicular bots can overwhelm the legitimate advertisements faster and more effectively.

3.2.3.2 Congestion Attack Mechanism

The attack starts with targeting a road on which to cause the congestion; this is given as a parameter to our simulator at configuration time. The parked bot cars are scattered around the map, inactive and parked. There do not need to be a lot of parked bot cars for our attack to succeed since the attack does not require their existence or continuous activity; especially for smaller grids/maps, we observed that practically no parked bot cars are necessary to execute a catastrophic attack. They turn on only when there are not enough compromised cars to cover the targeted area—the number of compromised vehicles needed is dynamic, and based on the size of the area, wireless ranges of these vehicles and their mobility patterns. Then, the parked bot cars move towards the targeted area, cover the uncovered parts, and begin advertising malicious congestion information.

From Manhattan-grid experiments, we observed that choosing a popular road to attack gives the best results. The attack is designed to be performed on only one road at a time, yet the experiments showed that the heavy congestion on the targeted road caused by our attack also spreads over to the neighboring roads, making them unusable as well.

Vehicular bots are allowed to create and advertise congestion information for roads that they have not been on. Therefore, this manipulative high coverage of their congestion responses causes the bogus measurements to disseminate and dominate very quickly. However, they stay under the plausibility filters' thresholds to be credible while still being manipulative. Vehicular bots also consider their distances to the roads that their bogus congestion measurements belong to; when vehicular bots lie about the measurements' timestamps, they need to be old enough to be plausible while still being fresh enough to be prioritized by other cars. Also, vehicular bots can respond to different cars with different congestion information based on the location of each requester in order to better manipulate each victim car into choosing a route that passes over the targeted road.

Exploiting Basic Safety Messages Cars broadcast their current speeds in their BSMs. These speed exchanges are used by each car in the congestion avoidance mechanism to calculate the total average speed on each road that it travels on. These average speed measurements are then used for travel time calculations and least congested route selections of other vehicles. The goal of our attack is to redirect as many cars as possible to the targeted road, eventually decreasing its average speed. A vehicle will not pick a route that contains the targeted road if it hears overwhelming reports of low speeds from the cars that actually went over that road. As a result of the congestion avoidance mechanism, the other victim cars that are considering to use the targeted road will start rerouting themselves, which will prevent the congestion from becoming heavier.

To overcome this effect, vehicular bots will advertise high speeds in their BSMs while traveling on the targeted road. In other words, they will continue to beacon that they are moving fast on the targeted road that is, in fact, heavily congested. This will exploit speed averaging mechanisms of the cars on the targeted road, causing their calculated averages to increase in order to help the effect of our attack continue as long as possible.

Exploiting Congestion Response Messages Vehicular bots carefully tailor each malicious congestion response based on the content of the corresponding congestion request. Manipulation strategies of the vehicular bots will be different depending on the requester. There are two different scenarios: the requester has the targeted road in its candidate routes, or it is not considering the targeted road at all.

In Figure 3.1, the victim Car A is approaching the targeted road from the bottom left side of the map. Its destination is marked with a star with the label "A" located at the right side of the map. The targeted road is contained in at least one of the car's candidate routes, meaning that we can potentially lure Car A into the target. The congestion responses from the vehicular bots will then blacklist any road that conflicts with their interest, i.e., any road that serves as alternative to the targeted road or gets the car away from it. All the roads that are parallel to the targeted road are considered as the direct alternatives. By advertising low average speeds for such roads, the vehicular bots ensure that the victim car is left with no choice but to take the targeted road on its way to the destination. We further improve our attack by making the vehicular bots advertise that all the roads that get the victim closer to the targeted road have high average speeds, and the ones that get the victim further have low average speeds. The way that the vehicular bots advertise the roads as congested or not congested is demonstrated in Figure 3.1.

Another interesting characteristic of the attack can be explained by including the victim Car B in the scenario in Figure 3.1. Consider that right after Car A chooses its route based on the received congestion responses, Car B approaches the targeted road from the top left side of the map. Even if all the vehicular bots are out of the

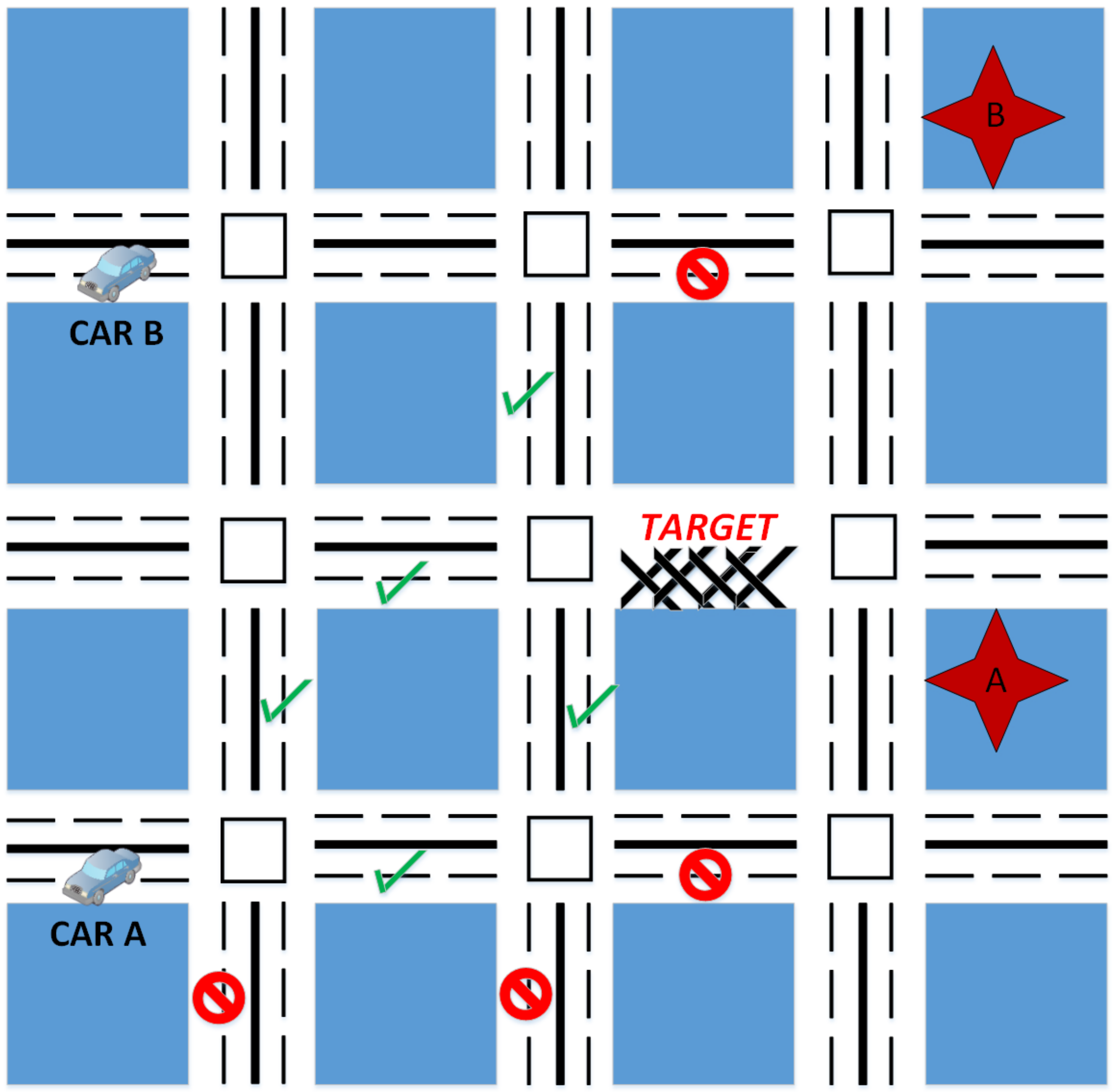


Figure 3.1: Malicious advertisements for the cars considering to take the targeted road

communication range of Car B when it sends a congestion request, Car A will be able to respond to Car B with all the bogus congestion information it has just received from the vehicular bots. Hence, the malicious advertisements sent by the vehicular bots in order to trick Car A will also exploit the route selection of Car B, even if there is no vehicular bot within its wireless range. We designed and specifically optimized the algorithm that creates bogus congestion measurements so that the previously disseminated information can still influence other victims as well.

Assume that the victim Car C in Figure 3.2 is not considering taking the targeted road for any of its candidate routes. The vehicular bots will then push Car C towards

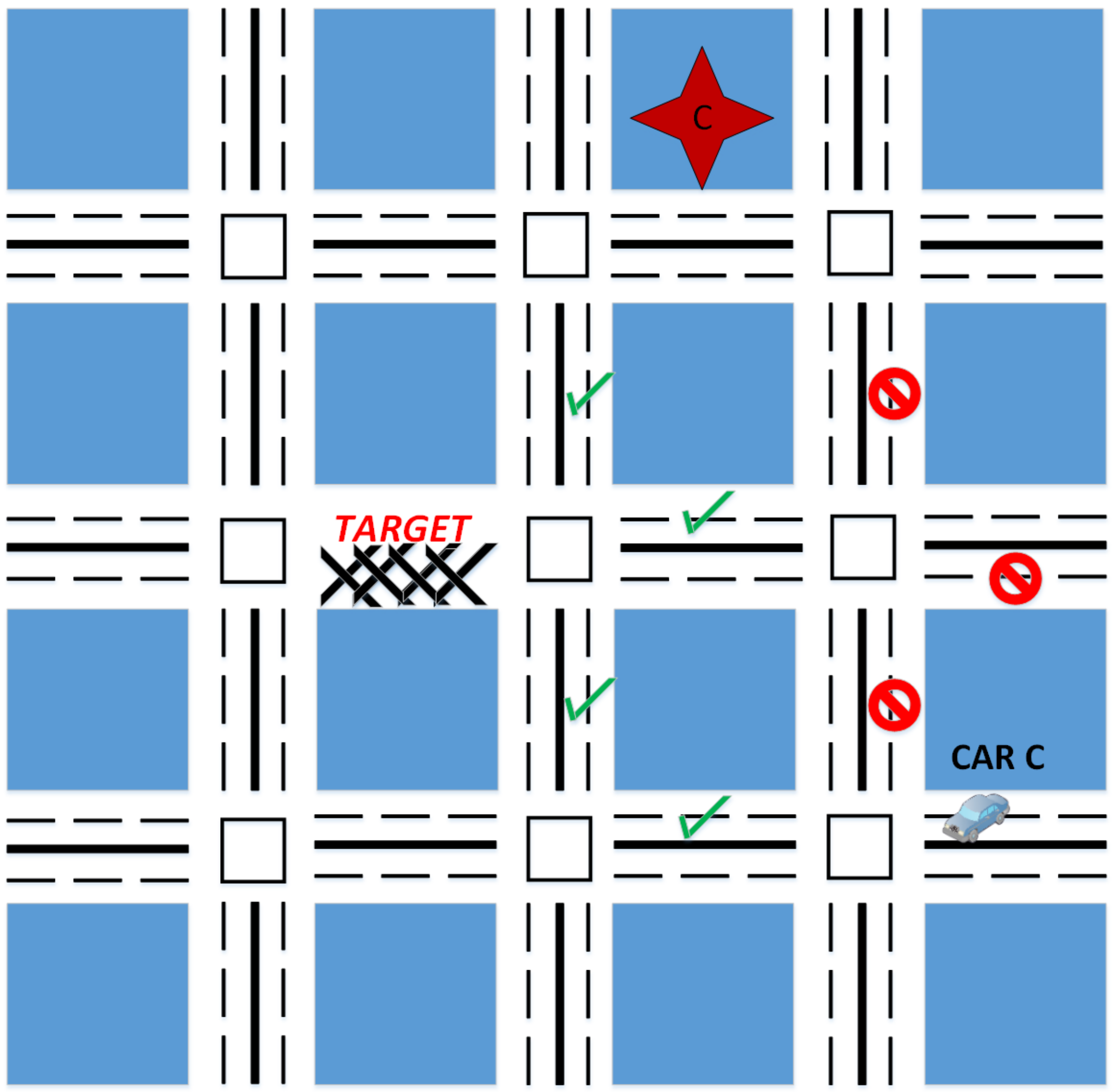


Figure 3.2: Malicious advertisements for the cars that are not considering to take the targeted road at all

the end of the targeted road as much as possible by advertising high average speeds for the roads that get the victim closer to the end point of the targeted road, and low average speeds for the ones that get it further away. Using the same method, if the vehicular bots push as many cars as possible towards the end point, the vehicles that try to exit the targeted road will have to wait for these cars to pass first before moving forward or making a turn. Therefore, the cars currently on the targeted road will not be able to leave fast, so the existing congestion on the target will increase even more. In addition, a vehicle cannot create a congestion measurement for a street before exiting that street—it would otherwise be an incomplete and inaccurate measurement. Since the speeds on roads would be suffering from the congestion created by our attack, the cars would not be able to complete their exits from roads as often. Hence, fewer legitimate congestion measurements would be created, while the vehicular bots could further increase their domination in terms of the quantity (and possibly, freshness) of their malicious measurements.

3.2.4 Evaluation

We evaluated our congestion attack over a timeframe of 2400 seconds in each simulation, which takes approximately 30 hours to complete. During this simulation period, we were able to test our attack with 2400 automated vehicles entering and leaving the map. We ran experiments with different percentages of vehicular bots over the total number of cars (1%, 5%, 10% and 20%). We randomized each car's initial and destination point at the beginning of each simulation to ensure that the congestion we observe is not simply due to the number of cars generated or possible bias in the predefined routes.

We focused on three metrics to show the success of our congestion attack: the average speed and the number of cars on the targeted road over time, and the overall trip time of each car. We assume that the decrease in the average speed and increase in the number of cars on the targeted road are strong indicators of traffic congestion. We used the overall trip times as a metric to show that our attack has a significant effect on both local and global traffic. We implemented our vehicular bots to start and stop the attack at predefined times, allowing us to compare the attack-enabled and attack-disabled scenarios and to measure how quickly the attack starts causing a significant amount of congestion.

As seen in Figure 3.3, our attack can cut the average speed on the targeted road significantly and quickly. In nearly 500 simulation seconds, average speeds drop 50% and follow a non-increasing trend line. In 1300 simulation seconds, the average speed reaches zero values, meaning that the targeted road is no longer usable. The negligible increases in the average speed after that point are due to the few cars that reach their destinations on the targeted road. However, the heavy congestion caused by our attack cannot be resolved despite these small increases in the average speed.

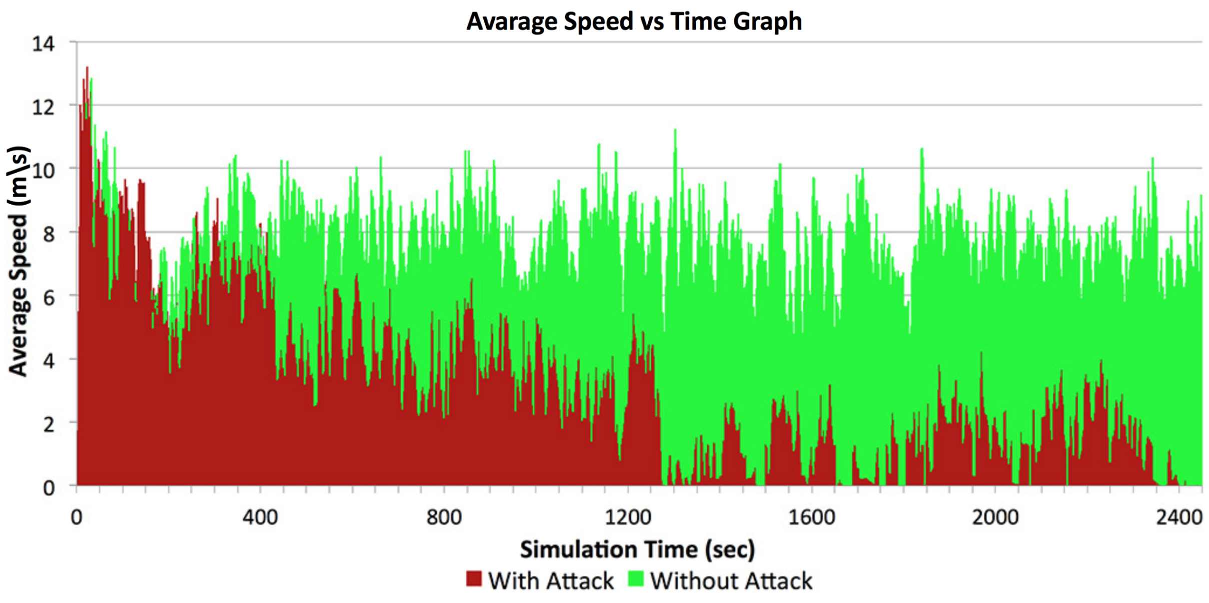


Figure 3.3: Average speed on the targeted road with and without active attack

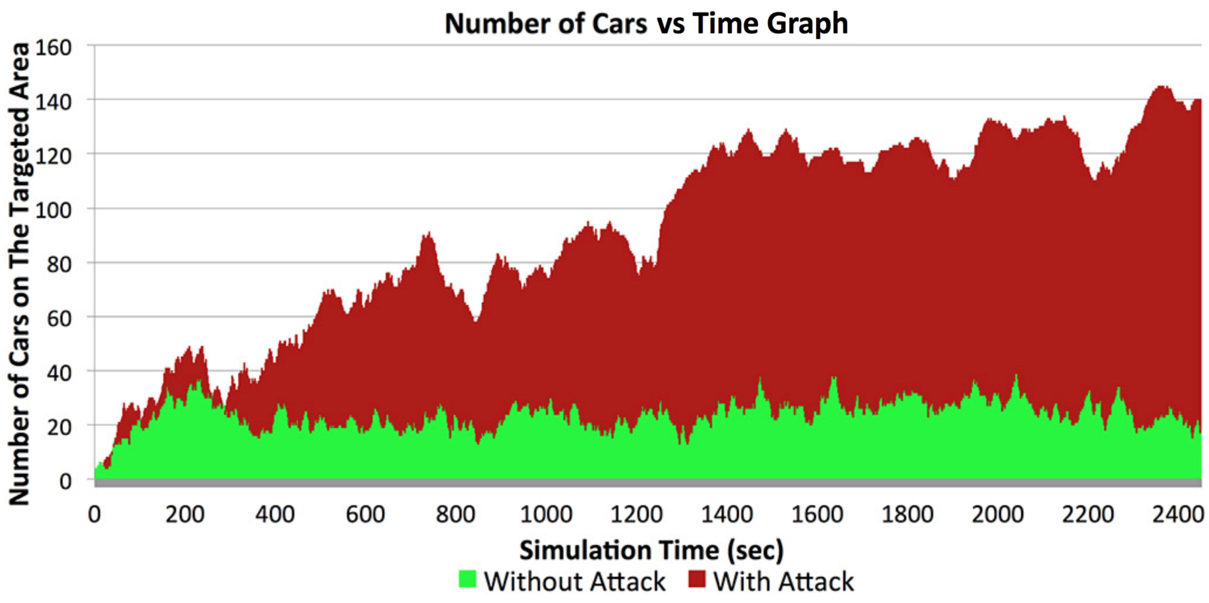


Figure 3.4: Number of cars on the targeted road with and without active attack

Figure 3.4 shows the success of the attack by comparing the number of cars on the targeted road from the attack-disabled experiment with the attack-enabled one. The number of vehicles on the targeted road increased by orders of magnitude for the latter case. Figure 3.4 shows that with our attack, at the end of the simulation, the number of cars on the targeted road is 7 times larger than the attack-disabled case. The cars on the targeted road cannot leave the road easily because our attack manipulates any cars that will not use the targeted road to pass through the intersection that the trapped cars have to use to leave. As a result, we observe a consistently increasing trend line in the number of vehicles, as shown in Figure 3.4, resulting in a highly congested road.

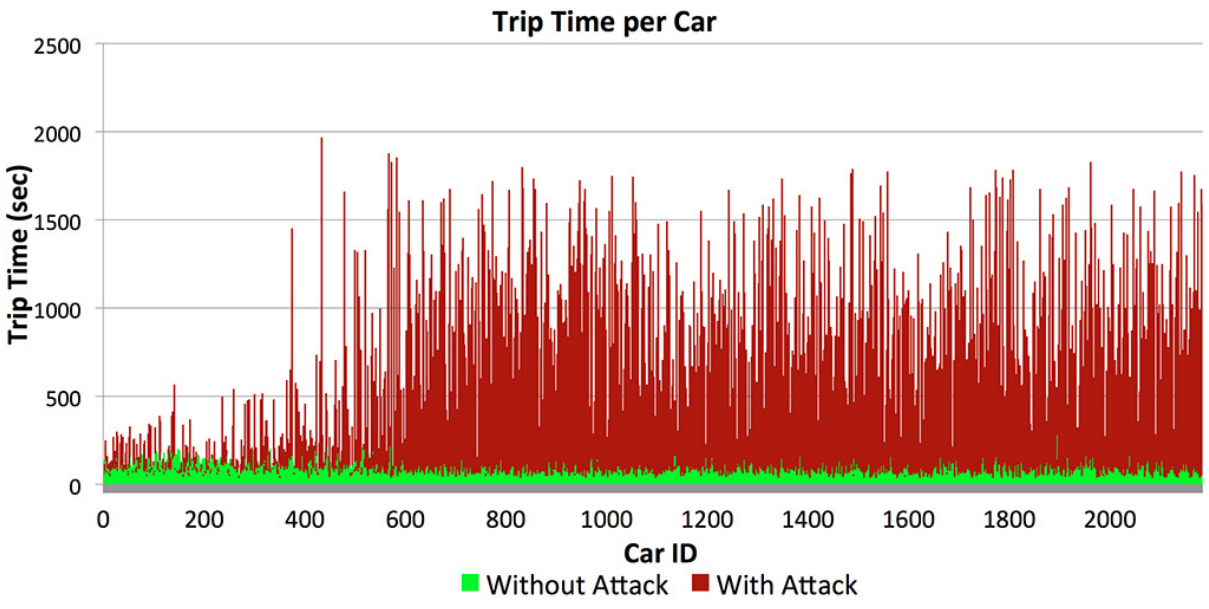


Figure 3.5: Trip times of all the cars on the map with and without active attack

In another experiment, we show that our attack impacts every car on the entire map, not just those around the targeted road. Figure 3.5 demonstrates the success of our attack in increasing the overall trip times of all the vehicles in the simulation. This experiment reveals that the local congestion caused by our attack also affects the cars not using the targeted road. For most of the vehicles, the trip times in the attack-enabled experiment are 10 times larger than in the attack-disabled experiment. In a real-life scenario, this result can be represented by an example where a car reaches its destination in 10 hours instead of 1 hour, which indicates a very severe congestion level.

Figure 3.6 demonstrates how fast the congestion caused by the attack on the targeted road expands to the surrounding roads. The graph shows the average speed on the targeted road, and the average of all the average speeds on the roads that are 1 or 2 hops away from the targeted road—i.e., on the roads adjacent to the targeted road or the roads adjacent to these roads, respectively. They follow the same decrease pattern in the average speeds as the targeted road. With around 200 simulation seconds delay between them, they all become completely unusable. While we only include a graph for 2 hops around the targeted road, we observed that the congestion also expands to the roads further than 2 hops in the same fashion.

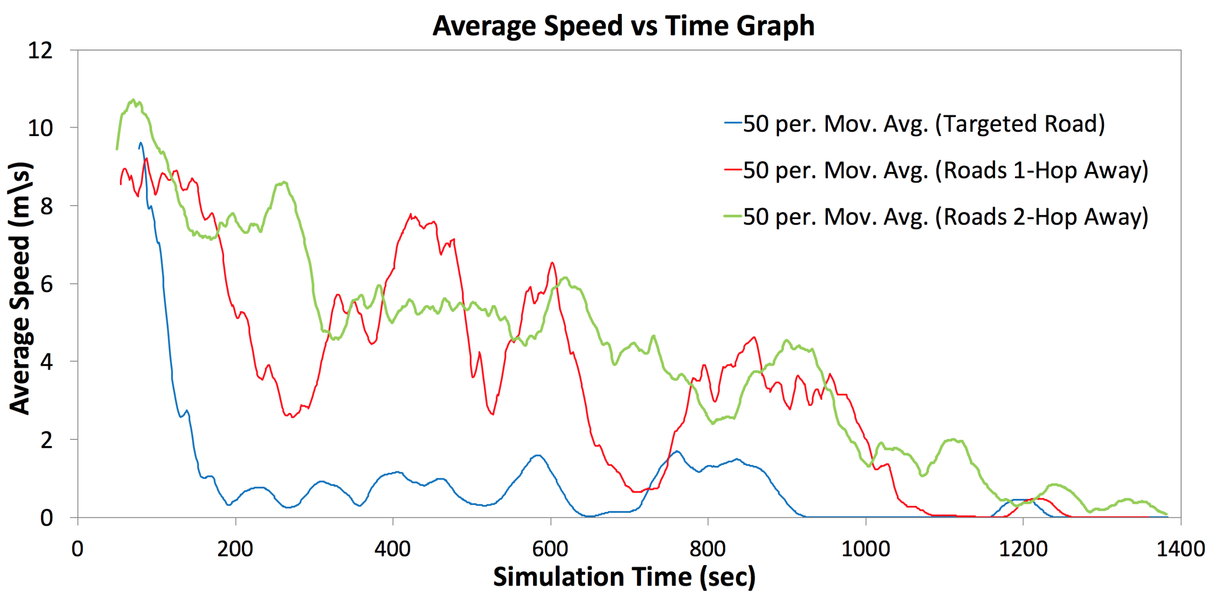


Figure 3.6: Average speed on the targeted road and neighboring roads

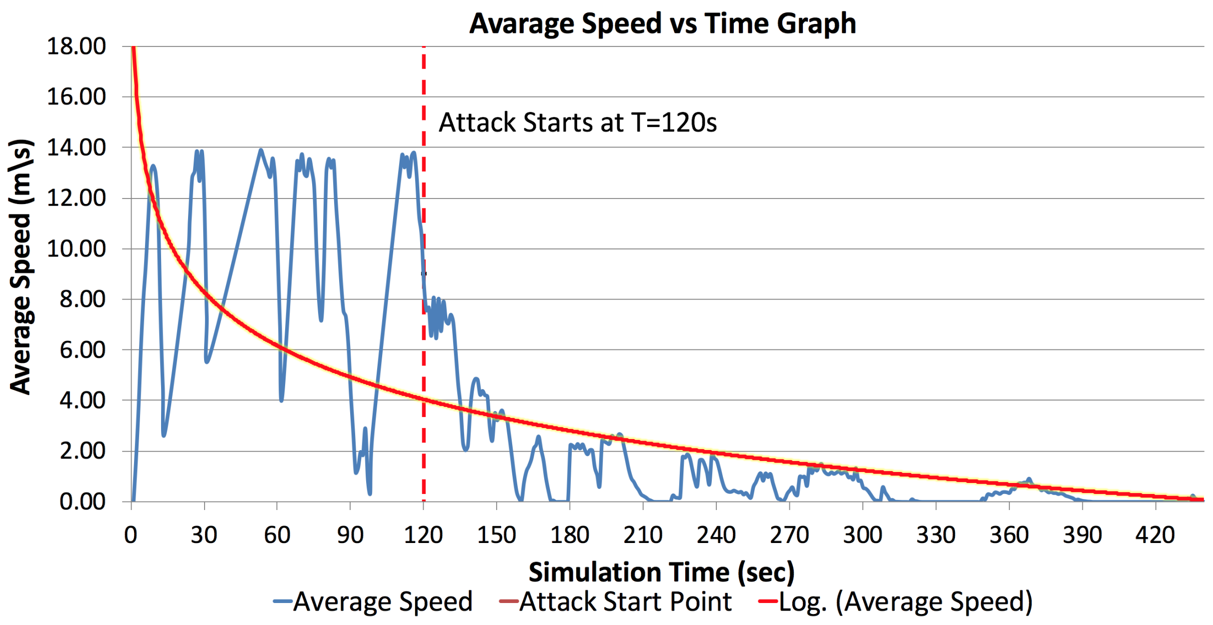


Figure 3.7: Average speed on the targeted road where the attack starts 120 seconds after the beginning of the experiment

The attack shows its effect very quickly. In Figure 3.7, the attack starts at the 120th simulation second, and in a very few seconds, we see the severe drop in the average speed on the targeted road. The graph follows a continuous decrease on the logarithmic trend line, drawn in Figure 3.7, until it reaches zero values. The speed of our attack can vary depending on how far from the beginning of the simulation we start the attack; the later it starts, the faster its effect will be observed. The reason is that at the beginning of the simulation not many cars are close enough to the targeted road yet. Our attack is really fast in luring the vehicles to use the targeted road, thus causing heavy congestion in just a few minutes.

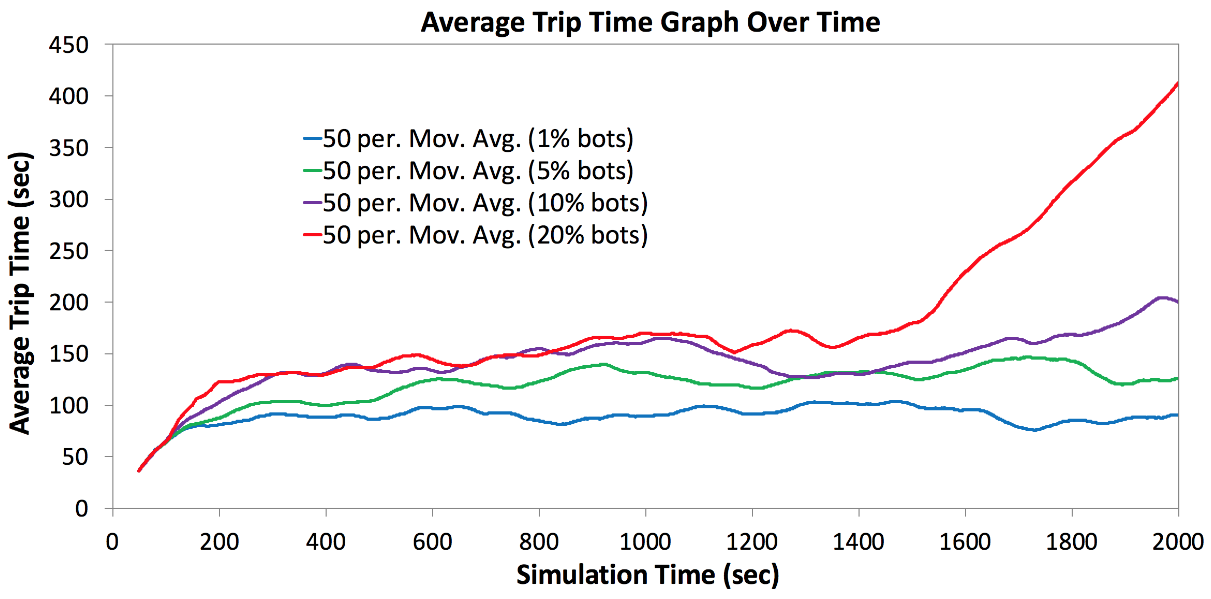


Figure 3.8: Average of all the cars' trip times for different percentages of vehicular bots

Let β be the probability of a car to be compromised and N be the total number of vehicles generated by the simulation; there will be $\beta \times N$ compromised cars and $(1-\beta) \times N$ uninfected cars created throughout the simulation. Figure 3.8 shows the effectiveness of our attack with different β values—1%, 5%, 10% and 20%. As seen in the graph, having even 1% of the cars around the targeted road in our vehicular botnet can increase the average trip time by 50%. With higher β values, the damage caused by our attack reaches even more severe levels; the average trip time grows exponentially.

3.2.5 Discussion

In this work, we attack the V2V reactive congestion avoidance request-reply scheme. This scheme is effective and efficient in terms of avoiding congestion in the absence of an attack. However, it is also particularly vulnerable because the car announces its candidate routes and the vehicular bots can pick their victims based on these routes. There are other schemes like [LMM11] in which the car does not request explicit information from other cars nor does it announce its routes. In [LMM11], the car waits for the congestion information to propagate to it via dissemination. The vehicular bots, in this case, cannot target a specific car but they will not have to anyway for our attack to be as effective, since they can still manipulate the congestion information propagated to the legitimate cars, as demonstrated in Figure 3.1. Besides, they could also create “tunnels” (by “blocking” certain roads) and lead unaware vehicles to congestion traps. As pointed out earlier, the cloud-based congestion avoidance schemes would be somewhat more resistant to our attack; however, they are much slower in reacting to and reporting congestion than the V2V schemes.

3.2.6 Conclusion

In this work, we presented the first vehicular botnet attack, namely a congestion attack, which can cause severe traffic congestion on any chosen road, making it and other roads surrounding it virtually unusable. We surveyed existing VANET congestion avoidance mechanisms as a possible target of our attack. We performed our attack particularly on existing V2V congestion avoidance schemes where autonomous vehicles completely rely on and follow computer generated routing instructions. To assess the effectiveness of the attack, we evaluated the traffic flows and delays when our attack was active and inactive. While the attack was performed on a simplified benchmark congestion avoidance mechanism, we argued that this type of attack applies to almost all V2V schemes, since in a V2V scheme cars draw the operational information from peers’ advertisements, which can be manipulated by vehicular bots. This work showed that vehicular botnet attacks can have physical consequences.

3.3 BOTVEILLANCE

VANET traffic safety applications rely on the location data broadcast in BSMs, allowing an attacker to track vehicles, which is a serious violation of the drivers’ privacy. Therefore, location privacy is a crucial security requirement for VANETs since the risk of being tracked might overshadow their advantages for drivers. [HGX10] shows that confidential information such as the home address of a driver can be identified by analyzing the routes taken by the vehicle. A surveillance attack can provide these routes to the attacker, which then can be used for criminal intentions such as tailing someone without any risk of getting caught due to close proximity, mugging, or identifying houses for burglaries.

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 [PFK14]. Public key cryptography is used to issue each pseudonym [SSB09]. Pseudonyms must be included in certificates and BSMs to provide authentication and non-repudiation [Wir10] [LLG15]. A pseudonym uniquely identifies a vehicle without revealing the identity of its driver to others. However, accountability is also required, to allow authorities to obtain the identities of misbehaving drivers using a pseudonym. While pseudonyms do not reveal true identities, a vehicle must keep changing its pseudonym to avoid tracking. Alternatively, a pseudonym could be assigned to a group rather than an individual vehicle [CPH07], which would provide location privacy without pseudonym change; however, addressing the accountability issue for group pseudonyms is difficult.

When a vehicle changes its pseudonym, all the other BSM information that could also uniquely identify the vehicle has to be changed as well. No pseudonym can be reused because if the old pseudonyms stay valid after the pseudonym changes, one can easily perform a Sybil attack with multiple valid identities [PSF15]. 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*, which contains the next pseudonyms to be used. The size of this set can be optimized as described in [LLZ08]. After the pseudonym set is depleted, a new set is obtained from the authorities.

The most important factor in the effectiveness of a pseudonym changing defense is its strategy of how, where and in what kind of situations pseudonyms should be changed—which is still a challenging research problem [BHV07]. A simple pseudonym changing scheme is not sufficient to prevent tracking [WMK10]. We believe that the best existing pseudonym changing scheme is Swing [LSH06]—the defense we implement for the evaluation of our attack—in terms of effectiveness, feasibility and robustness.

In this work, 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. BOTVEILLANCE uses geographically scattered vehicular bots for tracking the victim car. This would not be necessary if the victim itself was compromised; yet, compromising some vehicles might not be possible. The cooperation among the vehicular bots enables them to cover more ground with their combined communication ranges, thus 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, which makes BOTVEILLANCE adaptive to unexpected conditions and mobility patterns. Our attack is effective against most pseudonym changing schemes; however, we only implement Swing to test our attack since we argue that it is the best existing scheme. We exploit immutable vehicle properties and our location prediction heuristics to achieve accurate pseudonym linking. Since vehicles might advertise ambiguous location data to prevent tracking [DK05], the vehicular bots use INTERLOC, our interference-resistant RSSI-based localization mechanism built into the attack, which can accurately estimate the location of the tracked car by using the signal strength of its broadcasts. This work also provides a comprehensive survey on location privacy in pseudonymous systems, pointing out the weaknesses of the existing pseudonym changing schemes and standards for future researchers. BOTVEILLANCE shows that vehicular botnet attacks can also violate the privacy of VANETs in addition to having physical consequences.

3.3.1 Related Work

3.3.1.1 Pseudonym Changing Schemes

Providing location privacy in VANETs is necessary to ensure broad acceptance and deployment of these networks. Therefore, there has been a significant body of earlier work on protecting the location privacy of drivers, particularly the issuance and use of pseudonyms. Since pseudonyms alone cannot provide location privacy, and they need to be updated, an effective pseudonym changing scheme is the key to defending vehicles against tracking attacks. [WMK10] states that a simple pseudonym changing scheme would not be sufficient to thwart a powerful all-seing observer. A simple scheme might be vulnerable to tracking through a simple Bayesian traffic analysis [BMC08]. Many pseudonym changing schemes have been proposed, which can be divided into six groups [PSF15] based on their main pseudonym changing strategy:

***Fixed Change Period*:** Vehicles change their pseudonyms in a fixed period. [ESG10] implements a time-slotted pseudonym pool as an example of this scheme to have offline pseudonym issuance support. The problem with this strategy is that the attacker can easily link the old and new pseudonyms of a vehicle and track it once she figures out the period, which is also trivial [IMT10].

***Random Change Period*:** Vehicles change their pseudonyms in random unpredictable periods. A detailed description and analysis of this scheme is presented in [PLF11]. Since it is very likely that only few vehicles, if not only one, will change their pseudonyms at the same time due to randomization making the overlap of pseudonym changes rare, it is still easy to identify the new pseudonym of the victim by mobility analysis.

***Silent Periods*:** This strategy is not an alternative to the others but rather a supplement. Whichever pseudonym changing scheme is deployed, based on this strategy, vehicles stop broadcasting any message for a chosen period of time after they change their pseudonyms. Silent periods make it harder for attackers to link the old and new pseudonyms of a vehicle by mobility analysis. Examples of this strategy are CARAVAN [SHL05], AMOEBA [SLH07] and uncertainty-aware path cloaking algorithm [HGX10]. Effectiveness of this strategy, on the other hand, highly depends on how silent periods are set; having the same silent period for every pseudonym change, for example, enables an attacker to link pseudonyms by simply predicting the waking time. [Eic07] states that each silent period should be calculated as a function of vehicle's mobility. However, if this function is known to the attacker, he could predict a vehicle's next silent period by using its last advertised mobility data, and thus link its pseudonyms. [BHW09] proposes vehicles to change their pseudonyms and stay silent as long as their speed is under a preset threshold. They claim that not broadcasting BSMs while moving slower than the threshold would not jeopardize traffic safety. On the contrary, [LPB13] clearly shows that vehicles being silent at critical places such as intersections—which is what will happen based on [BHW09]—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*:** Any pseudonym changing scheme that proposes vehicles change their pseudonyms on their own is vulnerable to tracking attack. 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, the attacker can still link the victim car's old and new pseudonyms by mobility analysis even despite silent periods. Therefore, pseudonym changing schemes in the density-based change group and the next two groups aim to maximize the probability that multiple vehicles in close proximity change their pseudonyms simultaneously. In density-based pseudonym changing schemes, a vehicle changes its pseudonym when the current density of neighbor vehicles is above a threshold [CVT09a] [CVT09b]. Other vehicles following the same strategy increase the probability that multiple vehicles close to each other will change their pseudonyms, thus making tracking attacks more challenging. However, not every neighbor might perform a pseudonym change whenever the density is above a threshold in order to prolong the lifespan of its *pseudonym set*. Moreover, in situations where the traffic is sparse and the density of neighbor vehicles is always under the minimum threshold, density-based pseudonym changing schemes would cause the victim not to change its pseudonym for a long time—if ever—and therefore enable tracking.

***Collaborative Change*:** [FRF07] proposes applying the concept of *mix-zone*, introduced in [BS03] for anonymity, 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. However, the mix-zone defense is vulnerable to an insider attack. Therefore, considering that each mix-zone has to be large enough to build a sufficient awareness of surrounding traffic, the probability of one of the vehicular bots being located in victim car's mix-zone is significantly high. Since the mix-zone itself might not prevent tracking in case of insider attacks, the effectiveness of mix-zone-based pseudonym changing schemes determine the level of location privacy that can be provided. [SSB09] proposes each vehicle changing its pseudonym while entering a new mix-zone, hoping that there are other vehicles entering it as well and changing their pseudonyms simultaneously. However, old and new pseudonyms are still linkable if there is only one car in the mix-zone changing its pseudonym at the time, or there is even a slight difference between the times multiple cars enter the mix-zone. Also, if there is a significant distance between these vehicles, it is still possible to link the tracked car to its new pseudonym through mobility analysis. [BHV07] analyzes the effectiveness of mix-zones and argues that the optimum frequency of pseudonym change depends on the characteristics of the mix-zone (size, location, number of entry points), which are difficult to determine in practice. [GG07] introduces the concept of *mix-context* where a vehicle decides to change its pseudonym when it finds a sufficient number of other vehicles with similar mix-context to change pseudonyms with them at the same time. However, scarcity in the mix-zones significantly limits the success of such schemes. There is also a concept called *social spots* where mix-zones are created at crowded places such as stop signs, traffic lights, etc. In these social spots, vehicles change their pseudonyms collaboratively at the same time [LLL11], [LLL12], [LLS10]. This strategy also suffers from low privacy protection in low density scenarios [LL09]. Furthermore, it is not clear whether this scheme can be a standard since implementing the accountability requirement for social spots during pseudonym changes might be challenging.

***Vehicle-Centric Change*:** In vehicle-centric pseudonym changing schemes, each vehicle—independently and in a distributed manner—determines the best strategy to change its pseudonym. The best strategy might change depending on the current situation for each vehicle. For example, a vehicle would not be forced to change its pseudonym cooperatively with another vehicle if there are not many new pseudonyms left in its *pseudonym set*, or forced to wait without changing its pseudonym until the density of neighbor vehicles reaches some threshold. We believe that this group of schemes can provide the best privacy in the most realistic way since the same strategy applied in every situation, no matter how smart it is, will not provide everyone with the optimal privacy. Two vehicle-centric pseudonym changing schemes are presented in [LSH06]: Swap and Swing. In Swap, willing vehicles change their pseudonyms by exchanging them with each other with probability of 0.5 and then enter a random silent period. The idea is that, after the silent period, an attacker would not know which one is the tracked car since she would not even know if the exchange occurred. Unfortunately, exchanging pseudonyms with other vehicles introduces many problems related to accountability, and enables malicious vehicles to perform Sybil attacks by keeping the old pseudonyms active after each exchange. This can provide the vehicles with the ability to spoof every vehicle with which they exchanged pseudonyms. Swing, on the other hand, perform 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—collaborative pseudonym change, random pseudonym change period and random silent period—while being realistic and light-weight (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 3.3.2.2.

3.3.1.2 Tracking Attacks

The design of tracking attacks that can target these various pseudonym changing 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.

[LLX15] presents a tracking attack that creates trajectory predictions through a matrix completion algorithm and use a Kalman filter technique to increase prediction accuracy. However, the concept of pseudonyms, pseudonym changing schemes or any other location privacy methods were not considered and implemented for the evaluation of the attack. Therefore, the attack is ineffective when even a simple pseudonym changing scheme is deployed. [CBK06] also does not take any location privacy method into account, assuming all nodes use the same identifiers at all times. Tracking attacks that are not tested with any pseudonym changing scheme can be considered as simplistic and outdated. Also, [CBK06] assumes one all-seeing attacker node that performs tracking without explaining how a single node can obtain this complete knowledge of the network.

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

[BAG12] implements a simple tracking attack to evaluate the effectiveness of mix-zones in a real-life setting; however, it requires the deployment of sniffing stations, and therefore operates in a limited localized area. [PBF15] also depends on the installation of sniffing stations for tracking in a confined local area where it is realistic for an attacker to deploy them. The authors introduce a graph-based approach to determine the strategic places for deployment so that the number of sniffing stations required is minimized, without the purpose of designing a global-scale attack. [SLL10] mentions the possibility of an attack, which uses 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 [Ros13] and [Smi13], 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 hardware being deployed by the attacker is bound to be confined to a local area. Our attack, BOTVEILLANCE, 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, and resilient to location obfuscation methods due to its built-in RSSI-based localization mechanism.

[WSD14] mentions the context-linking attack conceptually without proposing a complete tracking attack. It does not suggest which context could actually be used for pseudonym linking or how to keep a victim under surveillance continuously. [HYM05] mentions global passive adversaries only as a possibility without any design suggestions. The authors use a correlation attack to link the old and new pseudonyms of a victim only by its predicted mobility, which is more simplistic than the correlation part of our attack. Moreover, they argue that the silent period defense mitigates the correlation attack, which our attack shows to be incorrect. [SLH07] considers that an insider node in a VANET can be compromised but it does not take into account that the compromised nodes can be organized or the powerful attacks this organization can enable. The authors assume that a single adversary can hear every BSM being exchanged without proposing how to do so given a limited communication range. They describe a context-linking attack similar to ours, taking random silent periods and mobility analysis into account; however, linking to the tracked car is performed either randomly or statistically, which is more simplistic and less accurate than ours. [GG07] implements a simple multi target tracking attack—again by an assumed global attacker—where only the mobility analysis is used for context-linking and tracking fails if there is more than one potential tracked car after the estimation, which will result in highly inaccurate tracking. Also, the only defense implemented is the random period pseudonym changing scheme without silent periods, which ignores improved pseudonym changing schemes.

All the earlier work on identifying possible attacks on location privacy assumes a global attacker without any suggestions on how this omniscience can be achieved. BOTVEILLANCE is the first implementation of such a global-scale attack with a thorough consideration of the most effective and at the same time realistic pseudonym changing strategies.

3.3.2 BOTVEILLANCE Design

In this section, we present the design details of BOTVEILLANCE—the immutable vehicle properties and location prediction heuristics to link pseudonyms, mechanism to choose the best tracker bot for adapting to changing conditions, and how it achieves the all-seeing attacker model and long-range global-scale tracking without requiring any additional hardware. As part of the attack mechanism, we discuss its novel built-in localization mechanism—INTERLOC—against defensive ambiguous location data advertisements. We also describe Swing as a defense and its configurations to evaluate our attack.

3.3.2.1 Identifiers to Change Beside Pseudonyms

BSMs contain information in addition to pseudonyms that could also be used as identifiers by attackers to track a vehicle. Therefore, it is widely accepted that these identifiers (MAC address, IP address, etc.) will need to be changed along with the vehicle's pseudonym. Naturally, 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 in this work that it could be exploited as an identifier if the area where the tracked car is being searched is small. The reason is that the vehicle size information in BSMs has a fine granularity (in centimeters) [Wir10], and we observed that 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 one of our attack heuristics for linking the old and new pseudonyms of the tracked car. Vehicle dimensions 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 3.9: Top 10 most popular car models in the US with their dimensions

The table in Figure 3.9 shows the top 10 most popular car models in the US with their percentages and dimensions [Cai17]. 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. Since we observe this for other car models as well, we implement only the models and dimensions in Figure 3.9 for the evaluation of our attack. The model of each car in the simulation is chosen randomly, where each model has the probability proportional to its popularity, to test the accuracy of our attack heuristic with real-life data.

3.3.2.2 Swing

Swing is a vehicle-centric pseudonym changing scheme where an individual vehicle is allowed to alter its strategy according to the current situation. Essential features for a pseudonym changing scheme to be effective, such as collaborative pseudonym change, random pseudonym change period and random silent period, are implemented in Swing. Here we discuss these features, important parameters to Swing and the values we set for them. To better describe how Swing works, we first describe its parameters:

Minimum Pseudonym Change Period **($period_{min}$):** The minimum time that must pass before pseudonym change is allowed. It is set based on a trade-off between privacy level and scalability of pseudonym issuance.

Maximum Pseudonym Change Period **($period_{max}$):** the lifespan of each pseudonym, after which a car must change its pseudonym.

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 to a neighbor for it to be considered for collaborative pseudonym change. Changing pseudonyms together with neighbors who are further than $neighbor_{max}$ is regarded as insufficient to cause confusion to the attacker.

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

Figure 3.10: Values of the Swing parameters chosen to test BOTVEILLANCE

Figure 3.10 shows the values of the Swing parameters for the evaluation of BOTVEILLANCE. These values are chosen to favor privacy—regardless of its implications on pseudonym issuance scalability and 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 Packet (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 if they receive a CNG_N from any neighbor in the vicinity of $neighbor_{max}$ in order to change collaboratively. If a vehicle cannot perform a collaborative pseudonym change until $period_{max}$ after the last pseudonym change, 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 another second and check again. When the neighbors receive a CNG_N, only the ones with pseudonyms that are $period_{min}$ or older will participate in the collaborative pseudonym change; the others will just ignore the CNG_N. This strategy makes the prediction of the time of next pseudonym change infeasible for attackers. Also, the independent and distributed nature of pseudonym change decisions makes this scheme more light-weight than its alternatives and more 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. Collaborative pseudonym change followed by a random silent period provides a significant level of confusion against attackers.

3.3.2.3 INTERLOC

In addition to changing pseudonyms, vehicles might also obfuscate their true positions—by advertising location data as inaccurate as possible without endangering traffic safety—to prevent tracking [DK05]. Therefore, the vehicular bots use INTERLOC to validate the locations advertised by the tracked car in its BSMs. The most effective way to perform this validation is figuring out where the BSMs are sent from by using the physical features of their transmission signals, particularly Received Signal Strength Indicator (RSSI). INTERLOC is an RSSI-based localization mechanism that is resistant to extreme levels of interference and mobility. It continuously learns and adapts to heterogeneous and changing interference levels in the environment by using an interference-aware radio propagation model, which is an improved version of the model in [XYG06]. It does not depend on the existence of RSUs or any other stationary roadside infrastructure for localization. It only needs *observers*—vehicular bots—to collect the RSSI values belonging to the tracked car's BSMs and triangulate the tracked car based on these values.

Localization Algorithm Instead of estimating the exact point where a vehicle might be with an error, INTERLOC estimates a small area where the vehicle is certainly located without an error—unlike most wireless localization mechanisms. The purpose of this design choice is tolerating the slight and innocent differences between the advertised and true positions of the vehicle, caused by the noise in its GPS.

Each observer samples the RSSI values by listening to the BSMs that are continuously broadcast from the vehicle being localized and estimates its distance to the vehicle using the following formula:

$$d = d_0 * 10^{\left(\frac{P_T - P_R - PL_0}{10\gamma}\right)}$$

d is the estimated distance between the observer and the localized vehicle. P_T is the transmission power (in dBm) of the vehicle broadcasting BSMs and P_R is the power of the signal received by the observer. PL_0 represents the path loss $(P_{T_0} - P_{R_0})$ at the reference distance d_0. The initialization variables PL_0 and d_0 have to be set in the beginning before the formula is used and they have to stay the same afterwards. These variables configure the formula according to the general path loss of the environment according to the map size. Finally, γ is the coefficient that the observers continuously adjust to fine-tune the formula based on heterogeneous and changing interference levels in the environment. This way, we turn the radio propagation model—originally discussed in [XYG06]—into an interference-aware model that is resistant to extreme levels of interference.

After all the observers in the vicinity of the localized car estimate their individual d values, they are sent alongside the observers' current locations to the chosen observer for it to process this aggregated data. Time lags due to various delays (propagation, the observer selection, etc.) are resolved by this processing observer using the timestamps of the received measurements. This chosen observer is always the current tracker bot in our attack.

The chosen observer then creates a circle for each participating observer, where the centre point is the location of the corresponding observer and the radius is the d value

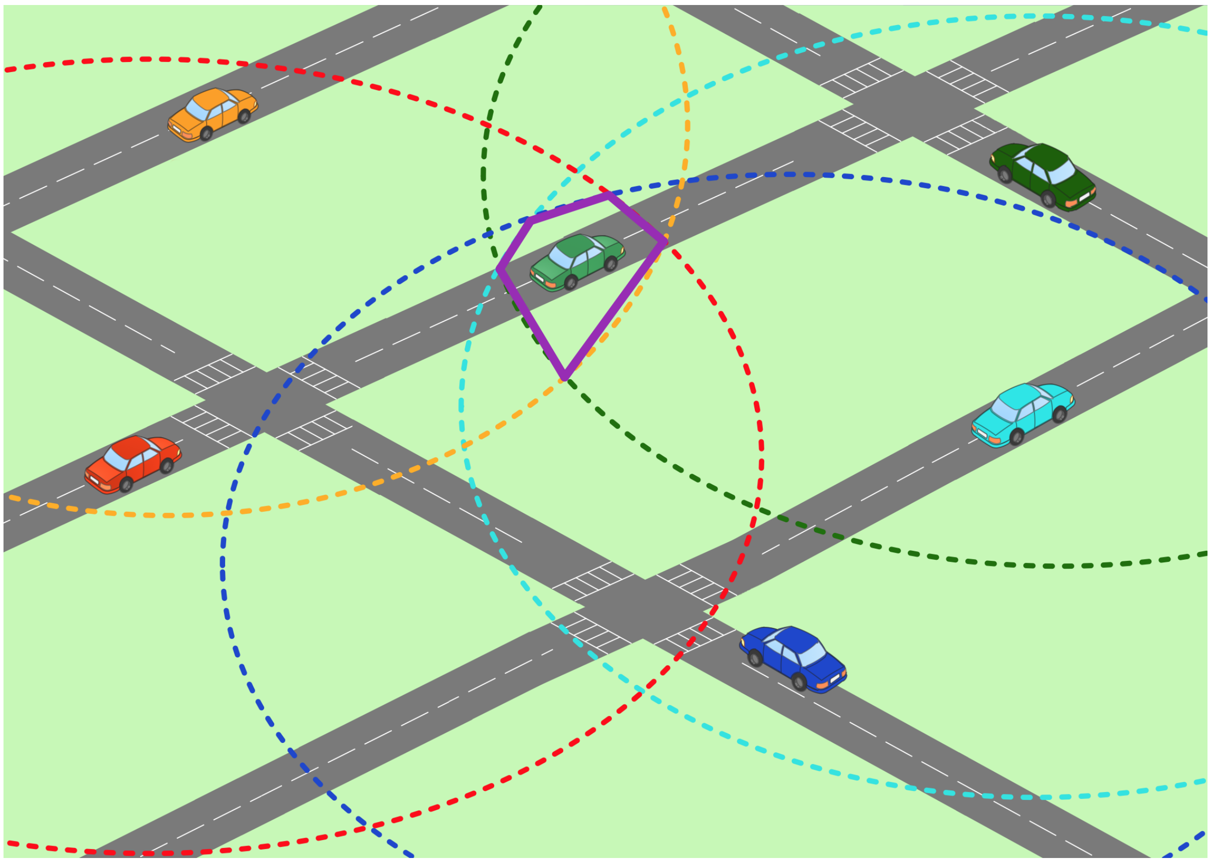


Figure 3.11: Localization performed by the observers and the estimated area

estimated by that observer. After all the circles are created, the chosen observer calculates the polygon whose corners are the intersection points of these circles, as demonstrated in Figure 3.11. The localized car will be inside this polygon without any estimation error, since each d will always be greater than or equal to the actual distance between the observer and the localized car. In an ideal environment without any interference in the RSSI values, the estimated polygon will just be a point on top of the localized car. As the interference levels increase, the RSSI values will get lower, increasing each d based on the interference measured by the corresponding observer. Therefore, the size of the polygon will get bigger but it will always contain the localized car inside due to the way d values are updated. That is why—by using INTERLOC—the vehicular bots can accurately validate the locations advertised by the tracked car in its BSMs.

Mechanism for Learning Interference Levels Learning and adapting to the changing interference levels are performed by exploiting the two variables of the radio propagation model: PL_0 and γ.

PL_0 is used to represent the initial overall interference levels before the localization can be performed. In [XYG06], this variable is set after a few iterations among a small number of stationary antennas. INTERLOC sets it after measuring power losses between every observer that is d_0 away from each other on the map, which enables sensing interference levels with a finer granularity and a wider range of angles. The value of d_0 is set in a way that maximizes the number of observers participating in the PL_0 calculation. As a result, INTERLOC represents the initial interference levels on the map much better and more thoroughly than the mechanism in [XYG06].

Learning the initial interference levels alone is not sufficient for achieving a high localization accuracy. Configuring only the PL_0 is considering interference levels to be the same everywhere on the map. However, interference levels differ based on where the localized vehicle is on the map and the angle between the vehicle and each observer. Also, the position of the vehicle being localized and the angle between the vehicle and each observer will continuously change. Therefore, we exploit γ in the radio propagation model to dynamically adapt to heterogeneous and changing interference levels.

Each observer has a set of (θ, γ) pairs, where each γ is calculated by sampling the interference level in the direction to another observer in the communication range and θ is the angle to that observer. Each observer recycles its set periodically due to the constant changes in positions and angles in order to adapt to the changing interference levels. Each (θ, γ) pair is calculated by the following formulas where the positions of the sampling and other observer are (x_1, y_1) and (x_2, y_2):

$$\gamma = \frac{P_T - P_R - PL_0}{10 \log_{10}(\frac{d}{d_0})} \qquad d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

$$\theta = \begin{cases} arctan(\frac{x1-x2}{y1-y2}), & \text{if } x_1 \leq x_2 \textit{ and } y_1 < y_2 \\ 180^\circ + arctan(\frac{x1-x2}{y1-y2}), & \text{if } y_1 \geq y_2 \\ 360^\circ + arctan(\frac{x1-x2}{y1-y2}), & \text{if } x_1 > x_2 \textit{ and } y_1 < y_2 \end{cases}$$

d is set to the distance between the advertised positions of the two observers and θ is the clockwise angle from the true north of the sampling observer to the other observer. An example of the θ calculation is shown in Figure 3.12.

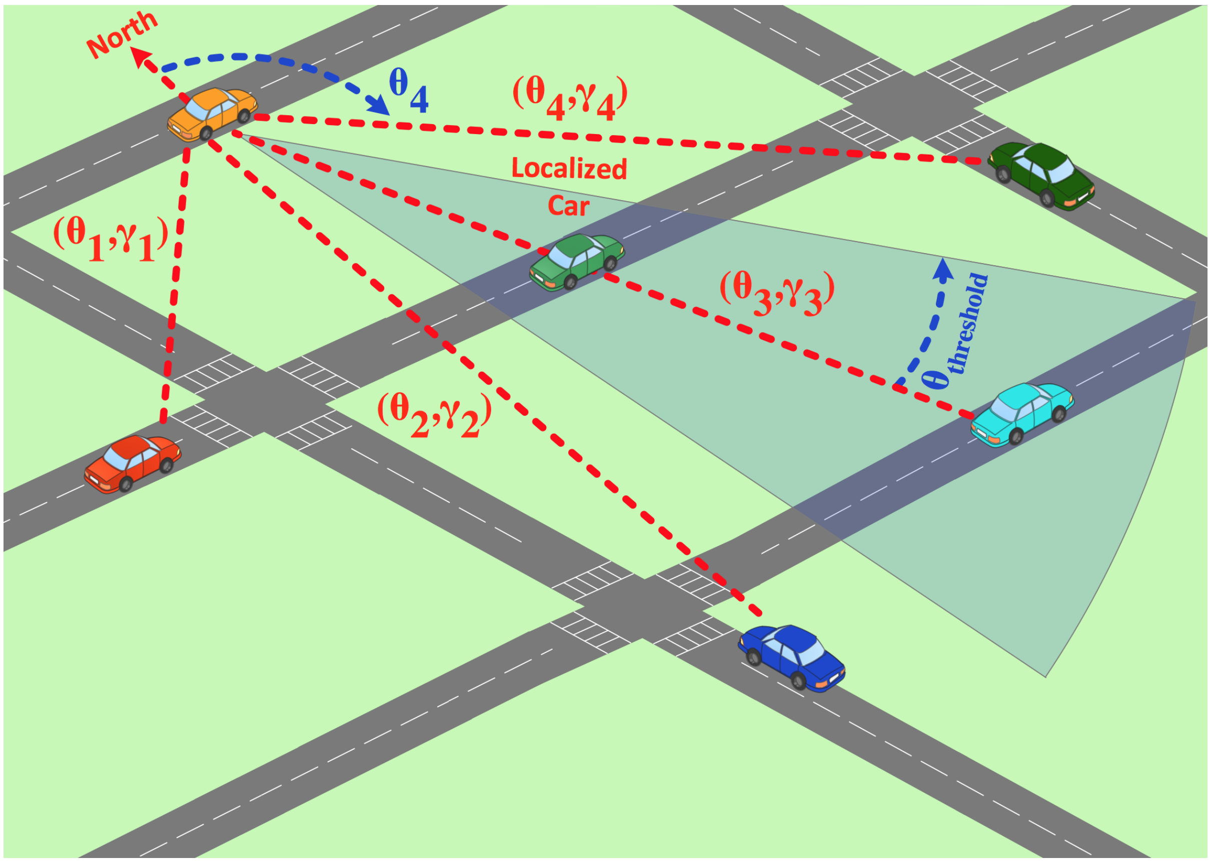


Figure 3.12: γ selection performed by each observer based on $\theta_{threshold}$ value

When a vehicle is to be localized, every observer in the vicinity of the vehicle attempts to choose the best (θ, γ) pair to use from its set, based on the angle to the vehicle being localized (θ_{loc}). Since θ_{loc} cannot be calculated without first knowing the estimated location of the vehicle, the initial estimation is performed by each observer using a default γ. Similarly to PL_0, the default γ is set to the value that minimizes the estimation error at setup time.

After the initial estimation of the vehicle's location by using the default γ, each observer then calculates its own θ_{loc} and determines the best (θ, γ) pair to use from its own set based on this θ_{loc}. The pair with the closest θ to θ_{loc} is regarded as the best choice $(\theta_{best}, \gamma_{best})$ and will be subjected to the final threshold test. If $|\theta_{best} - \theta_{loc}| \leq \theta_{threshold}$, then the γ_{best} will be used in the localization; otherwise, the default γ will be used until there is a θ_{best} that passes the threshold test.

Figure 3.12 depicts the γ selection mechanism, along with how an observer calculates θ using true north. In this scenario, γ_3 will be used by the sampling observer at the upper left since currently θ_3 is the angle that passes the threshold test. Every observer participating in the localization performs this γ selection based on its own set and can use different γ values than the other observers. This makes INTERLOC take the heterogeneity of interference levels into account for localization. Since the angles and locations of the vehicles are constantly changing, each observer periodically updates its γ value used in localization, along with periodically recycling its set with new (θ, γ) pairs, which ensures the adaptation to changing interference levels.

3.3.2.4 BOTVEILLANCE Initialization

The BOTVEILLANCE attack begins with the botmaster ordering vehicular bots to track the victim car. The order message must contain the initial 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. Since conventional botmasters are remote to avoid apprehension, it is likely performed over the Internet. Even though the Internet traffic has a chance of being detected, the infrequency of the communication with the botmaster and vehicles using the Internet also for other applications make it unlikely and worth the risk for protecting the botmaster. In case of detection, since only the tracker bot communicates with the botmaster, only one vehicular bot would be lost and another bot would become the tracker. When it comes to the communication among vehicular bots for cooperation during the attack, which is much more frequent, GHOST is used to avoid detection.

After the order is received by all the vehicular bots, they start the search for the vehicle with the initial 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} from any bot, 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, which is estimated through 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, because 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 the 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 very rare cases where the d_{tr}s of two candidates are equal, the comparison between the $pseudo_{cand}$s will break the tie. This way, the tracker bot will be chosen in a completely distributed manner since only one bot will not cease its tracking at the end of these comparisons. This is the initial tracker bot selection; continuous selections and tracking history handoffs are explained in the subsequent section.

3.3.2.5 BOTVEILLANCE Attack Mechanism

Tracker Bot's Attack State After the initial tracker bot is selected, it will initialize its *tracking state* ($state_{tr}$) and begin tracking the victim car. $state_{tr}$, a compound data structure, stores all the necessary information about the tracked car and the current tracking state. Any bot that receives it can easily 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. $info_{tracked}$ is used to estimate the position of the tracked car when it is in silent period. It contains the tracked car's pseudonym, speed, direction, dimensions, latitude and longitude at the time when the BSM is created, along with its timestamp ($last_{heard}$).

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 being 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. For these v_{est} updates, while the velocities of the tracked car are advertised and used only when it is not in the silent period, the velocities of the cars in front of it are used at all times.

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

Continuous Calculations of the Estimation Area Figure 3.13 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 random pseudonym change of the tracked car occurs. Since v_{est} continues being updated with the moving average of the velocities of the vehicles in front of the tracked car while it is in silent period, BOTVEILLANCE can still accurately estimate the mobility of the tracked car because it has to follow their mobility pattern one way or another.

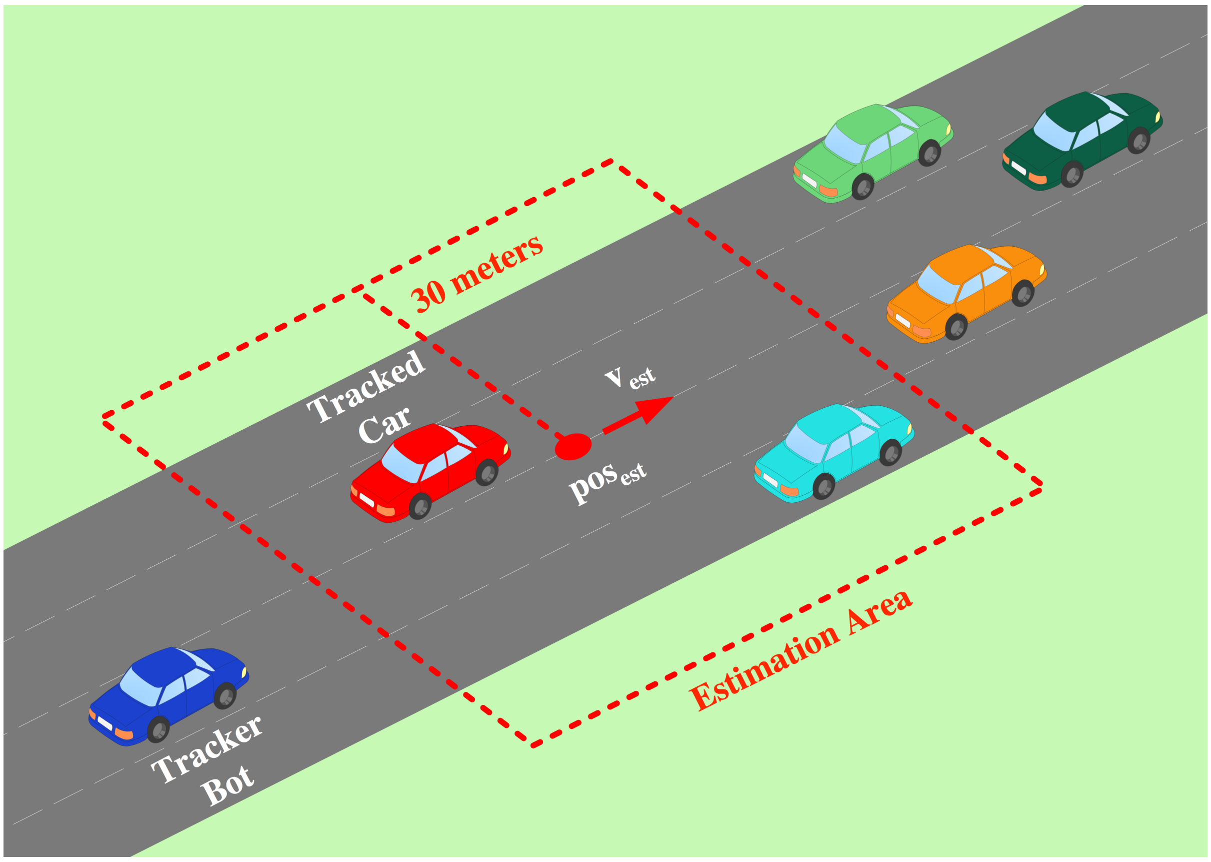


Figure 3.13: Tracking mechanism, v_{est} calculation and moving estimation area

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 regardless of whether the tracked car is in silent period or not since it ensures a very responsive context-linking once the silent period is detected. During the time when the tracked car's BSMs are being received without any break, the pos_{est} will then 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 approximate width of a highway in one direction is 18 meters (5 lanes x 3.6 meters for each lane [US14]), the radius of the estimation area is set to be 30 meters, as a design choice, to be robust to the ongoing turns and/or GPS errors of the tracked car. Every time the pos_{est} and corresponding estimation area is calculated, the tracker bot checks if the time elapsed since $last_{heard}$ has become $silent_{min}$. This is how the tracker bot detects if the tracked car entered silent period. 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.

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 of them is 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 too. If any of the attempts locates a vehicle with the same dimensions as the tracked car inside the estimation area, the context-linking will be complete, and the tracker bot will link the old pseudonym of the tracked car and the pseudonym of the vehicle in the estimation area. If there are multiple cars with the same dimensions as the tracked car in the estimation area, then the tracker bot will consider the vehicle that is closest to the pos_{est} as the tracked car. The radius of the estimation area has to be chosen carefully for a better context-linking accuracy; it should be large enough to contain the tracked car despite estimation errors, while also small enough to have as few vehicles with the tracked car's dimensions as possible inside. Finally, after the context-linking is complete, the $state_{tr}$ update mechanism will be configured with the tracked car's new pseudonym so that the $info_{tracked}$ and v_{est} 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.

Continuous Tracker Bot Selections In case context-linking keeps failing to locate the tracked car, 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 $silent_{max}$. This is when the tracker bot gives up, and initiates the next tracker bot selection and performs the $history_{tr}$ handoff.

The most important task that all bots perform during the tracker bot selection is searching in their communication range for the vehicles with the same dimensions as the tracked car. A tracker bot might lose a tracked car either because the mobility analysis moved the estimation area away from the tracked car, or because the tracked car woke up from silent period to make a turn but the tracker bot could not receive its BSMs due to packet losses and detect the turn. In other words, although the tracker bot could not locate the tracked car in the estimation area by any of its context-linking attempts, the tracked car might still be inside its communication range. Therefore, the tracker bot always participates in the selection of the next tracker bot as a candidate as well.

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 accordingly. Then, 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 which one is most likely to be the tracked car. Among these vehicles, the most likely tracked car will be the closest one to any one of the calculated pos_{est}s—providing the smallest $error_{est}$ as defined in Section 3.3.2.4. 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}—which has $error_{est}$, d_{tr} and $pseudo_{cand}$—except it also has the tracker bot's $state_{tr}$ inside so that the other bots can perform the search for the tracked car as well. 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 vehicle with the same dimensions as the tracked car—instead of setting a very high value for their $error_{est}$ 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 in their communication range 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 cease its tracking and withdraw from the competition. In case of any two $error_{est}$s being equal, the candidate with a smaller d_{tr} will win. If the d_{tr}s of the candidates that have the same $error_{est}$ are also equal, which is highly unlikely, the comparison between their $pseudo_{cand}$s will determine the winner since the pseudonyms are always unique.

Using $error_{est}$ as the most important factor in determining 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 tracker bot determines that it is not the best candidate to be the next tracker bot, it will send its whole $history_{tr}$ to the botmaster, which will append it to the tracking traces that have been sent by the previous tracker bots. In the beginning of the tracker bot selection, if the tracker bot does not receive a RES_{TR} from any bot and cannot locate any possible tracked car itself either, it will keep sending the REQ_{TR} until there is at least one potential tracker bot.

After the tracker bot selection is complete, the next tracker bot will reset the number of velocity samples for the v_{est} in its $state_{tr}$—which was received from the previous tracker bot—to one in order to favor the velocities advertised in the future in the v_{est} calculation.

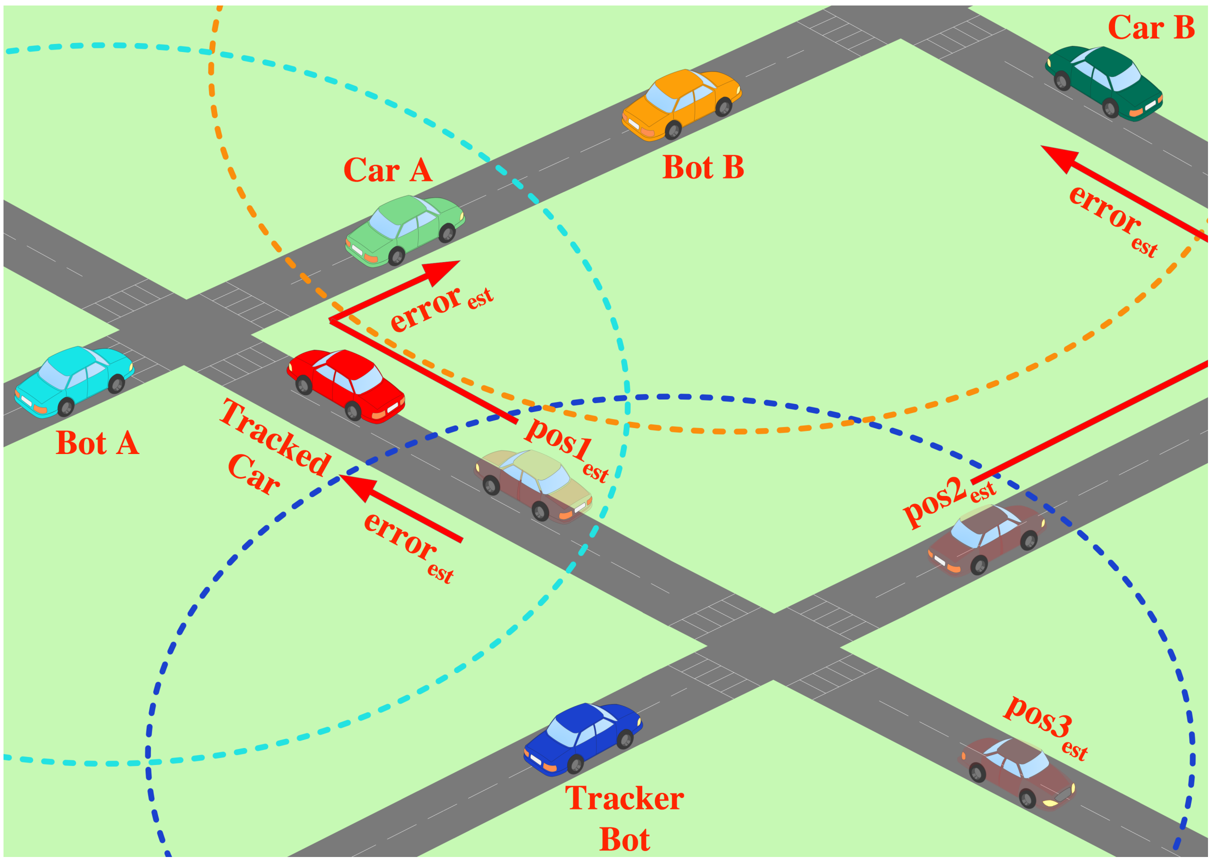


Figure 3.14: Calculation of multiple pos_{est}s due to an intersection and the tracker bot selection using the $error_{est}$s according to these pos_{est}s

Figure 3.14 depicts a tracker bot selection scenario where there are 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 calculates the $error_{est}$ for Car A the same way as Bot A does, it uses the $pos2_{est}$ to calculate the $error_{est}$ for Car B since it is closer than the $pos1_{est}$. Based on these calculations, to favor the smallest $error_{est}$, Bot A chooses the tracked car whereas Bot B chooses Car A as the most likely tracked car. After the corresponding RES_{TR}s are exchanged, Bot A becomes the next tracker bot due to having the smallest $error_{est}$ among all the tracker bot candidates.

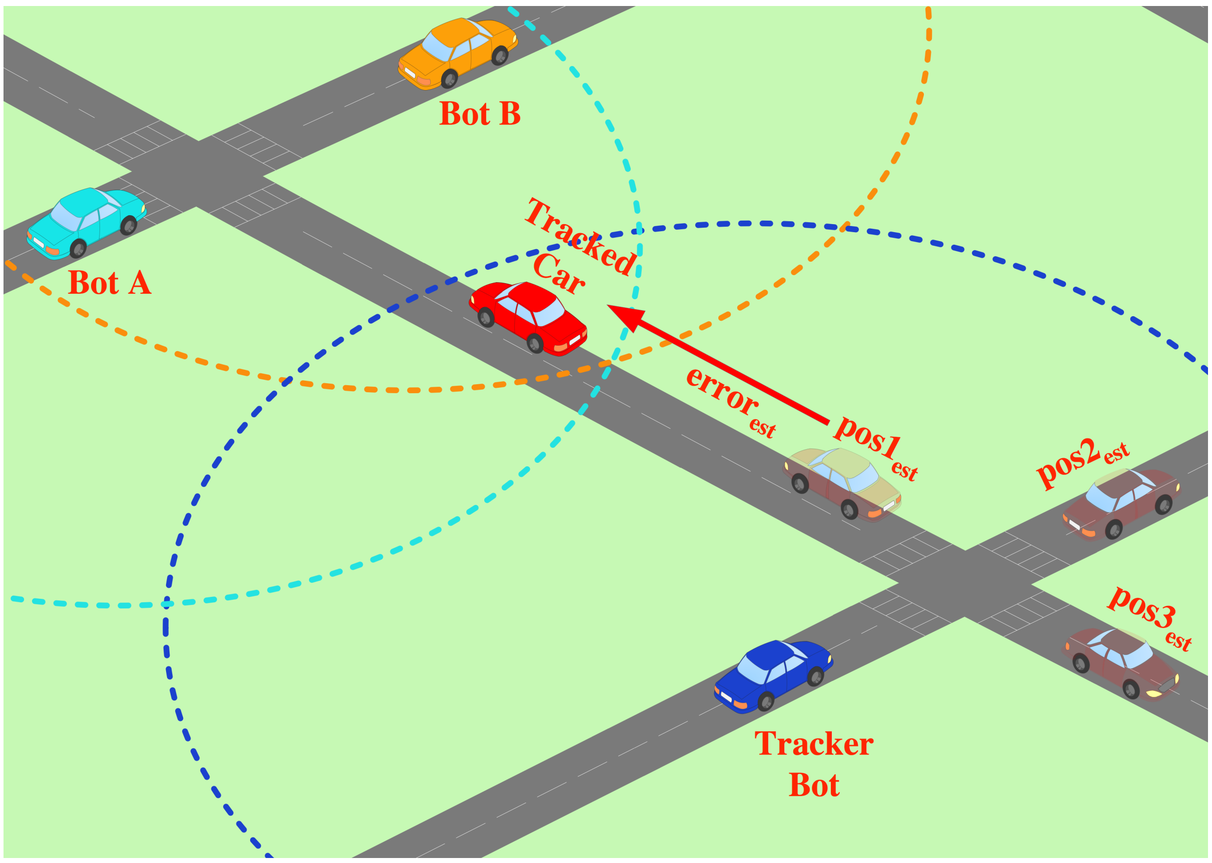


Figure 3.15: Tracker bot selection where the d_{tr}s determine the next tracker bot

Figure 3.15 shows a tracker bot selection scenario where 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 sends a REQ_{TR} with the actual values of its $error_{est}$ and d_{tr} and is a legitimate candidate in the selection of the next tracker bot. In fact, when the other candidates receive the REQ_{TR} and determine that they are worse candidates, they will not even broadcast a RES_{TR}. However, for the scenario in Figure 3.15, Bot A becomes the next tracker bot since it has the smallest d_{tr}, given that all the $error_{est}$s are the same. 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, in Figure 3.15, even though the tracker bot has the tracked car in its communication range, the tracked car is about to get out of it; therefore, using d_{tr} as a factor in the tracker bot selection enables the tracking to be handed off to Bot A, which is likely to keep the tracked car under surveillance longer.

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} so that it can be broadcast to every bot in the vicinity of all possible pos_{est}s. In this case, bots will not only broadcast their RES_{TR} to the other bots but also to the botmaster. If the botmaster does not receive any RES_{TR}, it will keep resubmitting 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, there were many cases in our simulations where 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—calculated based on the last v_{est} and all this time since the tracked car is lost—is low. That is how our attack always has a reasonable tracking accuracy even in the harshest conditions.

3.3.3 Evaluation

We subjected BOTVEILLANCE to rigorous and challenging experiments. As discussed in Section 3.3.2.2, Swing parameters are set to favor privacy. We used a Manhattan mobility model to maximize the number of intersections and possible turns in order to evaluate our attack in the toughest 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 $\approx$ 70 kilometers.

We ran 10 simulations for each value on the x-axis of each graph. We produced two accuracy metrics to evaluate BOTVEILLANCE:

***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 in this section is produced by varying each relevant attack or Swing configuration, whose effect is examined on the tracking accuracy—while keeping all the other configurations constant at their standard values. The standard values for these configurations in our attack are shown in Figure 3.16.

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

Figure 3.16: Standard values for the varied Swing and attack configurations

The standard values in Figure 3.16 are chosen to be realistic based on the VANET standards, traffic safety and pseudonym issuance scalability, rather than providing optimal tracking accuracy for our attack. They are set to more challenging values for each graph to test the robustness of BOTVEILLANCE to edge cases and investigate the correlation between each configuration and tracking accuracy.

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 3.17 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 percentage of vehicular bots over all the cars in the world. 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. During tracker bot selections, the distance between a bot and the tracked car is an important factor to determine the best tracker bot, minimizing the total number of tracker bots needed throughout the simulation by maximizing the probability of each tracker bot having the tracked car inside its communication range for a long time.

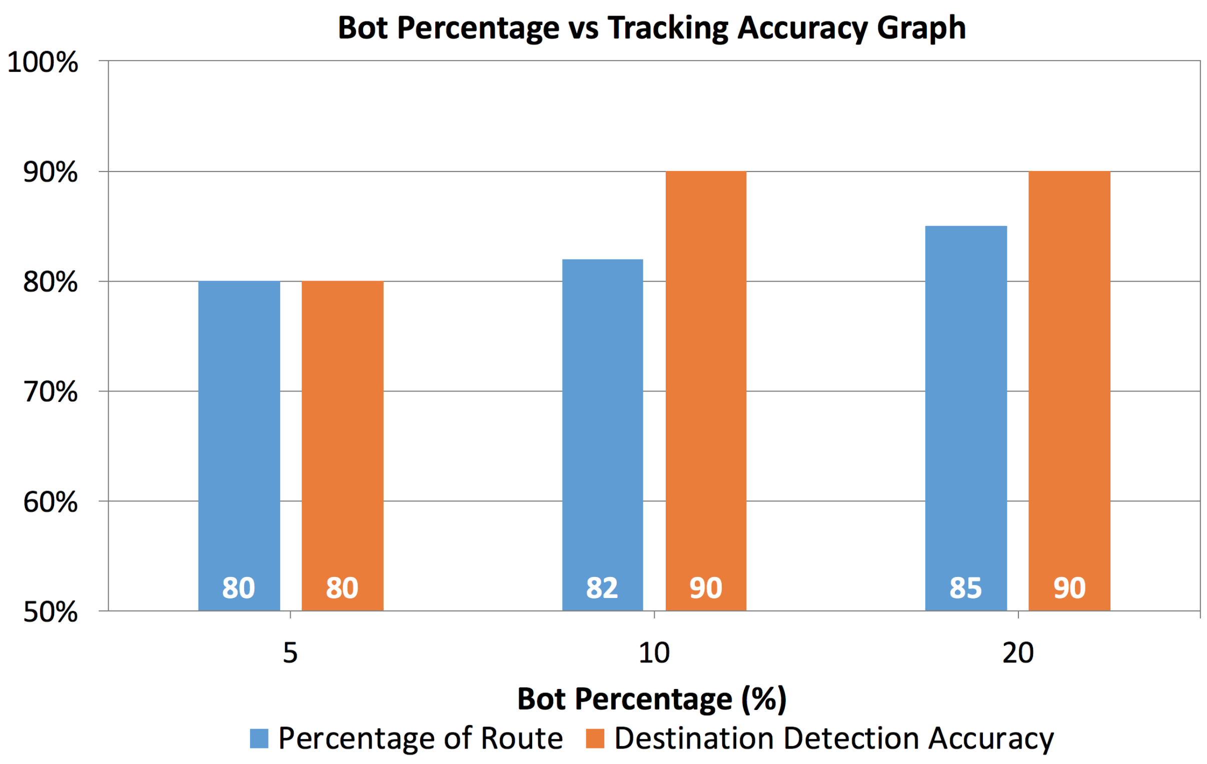


Figure 3.17: Tracking accuracies with different percentages of vehicular bots

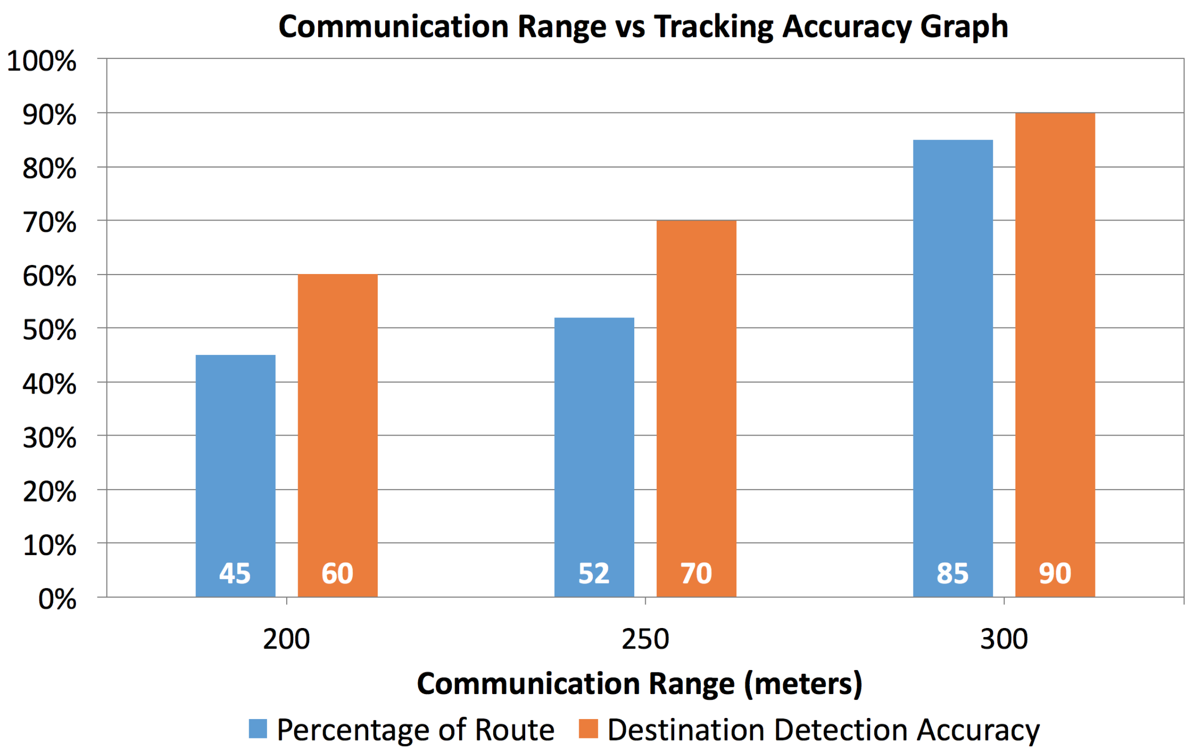


Figure 3.18: Tracking accuracies with different wireless communication ranges

Figure 3.18 shows the tracking accuracy for each wireless communication range set for all vehicles. The standard communication range for IEEE 802.11p is 300 meters [Wir10]. However, we evaluate the resilience of BOTVEILLANCE to smaller ranges since, in some cases, the effective communication range of a vehicle might shrink due to environmental factors like interference and obstacles. The experiment results in Figure 3.18 reaffirm the direct correlation between the tracking accuracy and size of the area covered by vehicular bots. In this case, the tracking accuracy changes exponentially with communication range due to the covered area being proportional to the communication range squared.

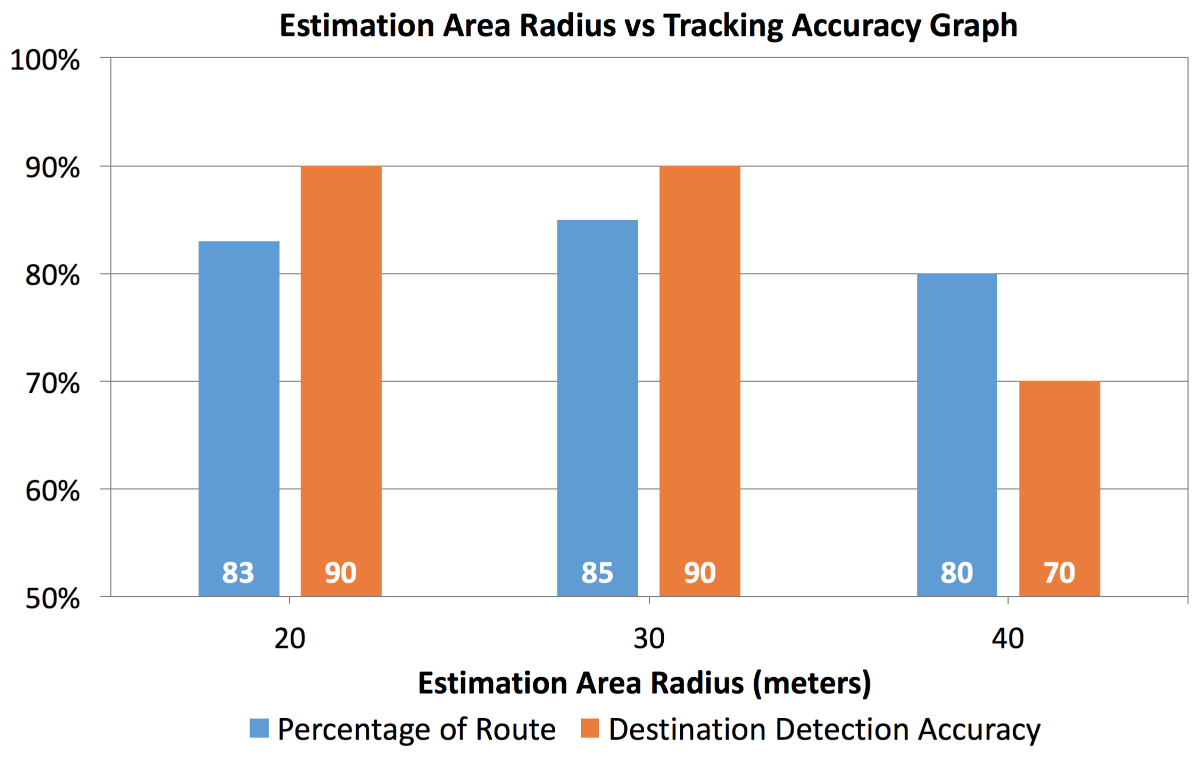


Figure 3.19: Tracking accuracies with estimation areas of different radii

The radius of the estimation area is also one of the factors that affect the tracking accuracy—particularly the accuracy of context-linking. A suboptimal radius increases the probability of the situation where 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 3.19 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 and minimum silence period, previously referred as $period_{max}$ and $silent_{min}$—on the tracking accuracy since they are already similar to the effects of the other parameters. $period_{max}$ has the similar effect to $period_{min}$ and $neighbor_{max}$, while $silent_{min}$ is similar to $silent_{max}$.

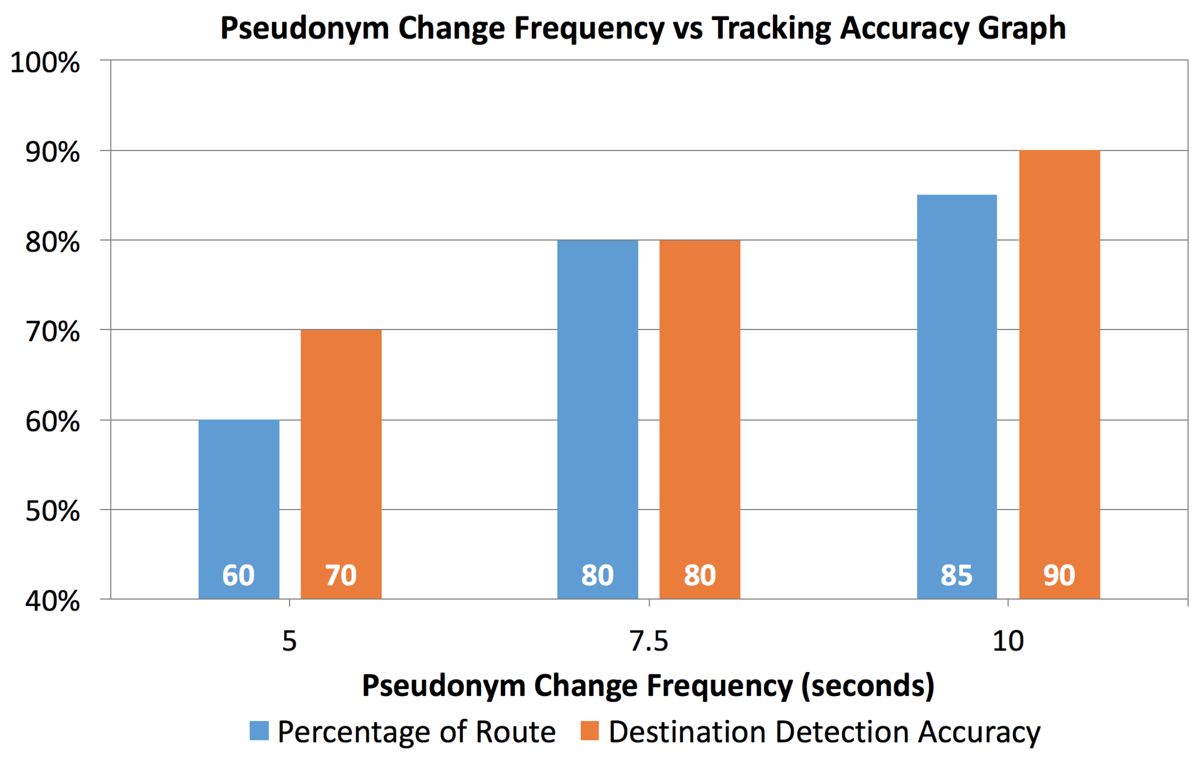


Figure 3.20: Tracking accuracies with different pseudonym change frequencies

Figure 3.20 shows the tracking accuracy for each $period_{max}$ set for Swing. The value of $period_{max}$ for our attack, 10 seconds, is already borderline unrealistic due to scalability of pseudonym issuance. Nevertheless, we still investigate the robustness of BOTVEILLANCE 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 $period_{max}$.

Silent periods provide a significant level of confusion among trackers; however, their values have to be limited due to the requirements for traffic safety, which is the primary goal of VANETs. Figure 3.21 shows the tracking accuracy for each $silent_{min}$ that enforces a lower bound for random silent period selections. Regardless of how unrealistically high the silent periods are, BOTVEILLANCE achieves a reasonable tracking accuracy due to

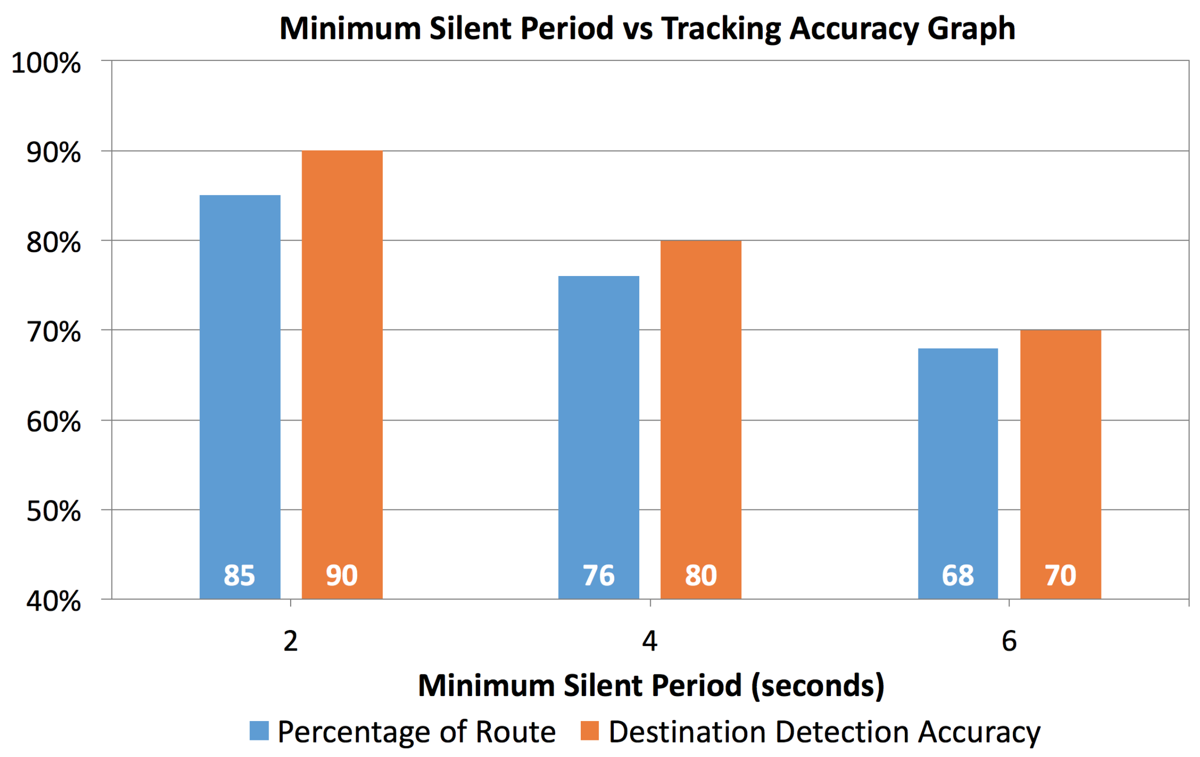


Figure 3.21: Tracking accuracies with different minimum silent periods

its effective context-linking heuristics and continuous tracker bot selections.

3.3.4 Conclusion

In this work, we presented BOTVEILLANCE—an adaptive cooperative surveillance attack performed by vehicular botnets—along with its novel built-in localization mechanism, INTERLOC. 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 provided a comprehensive survey on the existing pseudonym changing schemes and standards, with discussions of their weaknesses. 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. This work showed that vehicular botnet attacks can also violate location privacy—one of the most important security requirements for VANETs—in addition to being able to have physical consequences.

3.4 RIoT

The concept of IoT is expected to significantly improve many aspects of our lives, from Industry 4.0 and health applications to efficient workspaces and smart homes. Therefore, its popularity is growing more and more each day, along with its market share, supported by emerging applications and business models. The IoT market is expected to reach $1.2 trillion in 2022 [Col18] with 42.62 billion IoT devices worldwide [Sta16]. The security of IoT devices is often neglected by their manufacturers due to financial concerns and insufficient hardware capabilities of these devices, such as a limited battery capacity and computing power. Attackers are likely to come up with more and more exploits to compromise IoT devices as they become increasingly integral part of our lives [Mar18]; therefore, the security of these devices will soon be too important to overlook. However, due to their hardware limitations, the most feasible way to protect IoT devices would be outsourcing the task of preventing possible attacks to the firewalls installed on the gateway access points, through which these devices connect to the outside world. Such security measures, on the other hand, could protect IoT devices only from the attacks performed over the Internet, leaving their wireless ad hoc communication vulnerable. Therefore, an attacker could either directly use the peer-to-peer wireless connections to IoT devices if they support Simple Service Discovery Protocol/Universal Plug and Play Protocol (SSDP/UPNP) [BW15], or spoof the wireless access points through which they connect to the Internet [Hil18], or force them to revert to a known insecure communication mechanism [Tie18]—in order to deliver exploit payloads to them. Yet, given the number of IoT devices expected to exist, it would be infeasible for a single attacker to compromise them one by one this way.

In this work, we present RIoT—a rapid exploit delivery mechanism that can compromise a significant percentage of the IoT devices in an area of interest within a short amount of time, by taking advantage of the mobility and collective communication range of vehicular bots. It is the first attack in the literature against IoT devices using vehicles. RIoT takes the simple concept of wardriving [DER08] [SGA11] to a much more powerful level—using multiple cooperating vehicles for compromising IoT devices, rather than a single vehicle scanning for only wireless access points. This work does not present new exploits against IoT devices. Its novelty comes from the mechanism that can quickly deliver the payloads of existing exploits to the IoT devices in an area of interest, as well as from realistic experimentation with realistic traffic patterns and placement of these devices. RIoT's exploit database used by vehicular bots consists of several IoT vulnerabilities identified by earlier research. Identifying all possible vulnerabilities of every type of IoT device is out of the scope of this work and not necessary for demonstrating the effectiveness of our exploit delivery mechanism. However, RIoT's exploit database can easily be extended with new and possibly more effective exploits due to its adaptive design. While RIoT could also be used for constructive purposes such as penetration-testing a smart city, it is much more likely to be used for malicious intent. This work is important to show that vehicular botnets are threats not only to VANETs; they can also be used by attackers as an effective tool to perform powerful attacks against other important systems and networks.

3.4.1 IoT Device Categories

In order to accurately evaluate the effectiveness of our attack, it is important to determine the number and types of IoT devices to be used in the simulation, as realistically as possible. Therefore, we first identified the most popular IoT device categories that are expected to be on the market: industrial, consumer, medical, security, and retail [Int19]. We then determined the types of existing devices that are expected to be IoT-capable, under all these categories. Afterwards, we used the market sales data in the US in order to estimate the total number of IoT devices under each category, as well as the percentages of the categories. The types of these devices and their categories are temperature sensors [Fit13] [Nat15] [Pla19] and robotic machinery [GM17] for industrial; smart televisions [Ngu17], smart speakers [Per18] and smart refrigerators [Nav12] for consumer; wearable devices [Gru15] and health monitoring sensors [Ame19] [New17] for medical; smart security cameras [Vla09] and smart locks [Mar17] for security; and retail sensors [Pio17] and barcode scanners [ABI16] for retail. Obviously the popularity of particular classes of devices might change over time, but that factor does not substantially alter our results, and could be easily tested for by including newly popular device types in the simulation.

Figure 3.22 shows the expected number of devices under each IoT device category and

	Expected Number of IoT Devices	Percentage (%)
Industrial	122,337,619	36.88
Consumer	99,450,000	29.98
Medical	52,918,610	15.95
Security	32,000,000	9.50
Retail	25,500,000	7.69

Figure 3.22: Percentages of individual IoT device categories and their expected number of devices

the corresponding percentages. For realistic experimentation, at the beginning of each simulation, the IoT devices under each category are inserted into the simulation according to the percentage of the category. Existing vulnerabilities of the individual device types under each category comprise RIoT's exploit database: [MQP17] for industrial, [Cim17] and [Ley15] for consumer, [CS16] and [JS17] for medical, and [Pal17] for security. There are also vulnerabilities in the exploit database—such as [Fra17], [FBR16] and [Tie18]—that might affect every IoT device regardless of their types or categories. In order to test the effectiveness of our exploit delivery mechanism alone, during the simulation, we consider an IoT device as compromised if it just gets inside a vehicular bot's communication range so that the content of the exploit database or the specifications of each exploit (payload size, device version, etc.) do not influence the experiment results. The reason is that the content of the exploit database is expected to change anyway as the IoT standards, device types and their specifications change. Also for the same reason, assessing the effectiveness of individual exploits is out of the scope of this work and irrelevant with regards to demonstrating the effectiveness of our exploit delivery mechanism.

3.4.2 RIoT Design

3.4.2.1 Simulation Map and IoT Device Placement

Besides the realistic number and types of IoT devices used in the evaluation of RIoT, deploying an accurate city scenario and a realistic placement of these devices in it also play an important role in providing realistic experimentation. Therefore, we use a 24-hour Luxembourg scenario designed based on the real traffic patterns and volumes of the city. This scenario simulates the real traffic patterns throughout the day, and interactions with the road infrastructure, such as actuated traffic lights, with very high accuracy. Therefore, it enables us to evaluate RIoT with realistic mobility patterns on a scale as large as an entire city. The details of the Luxembourg scenario are described in [CFE15].

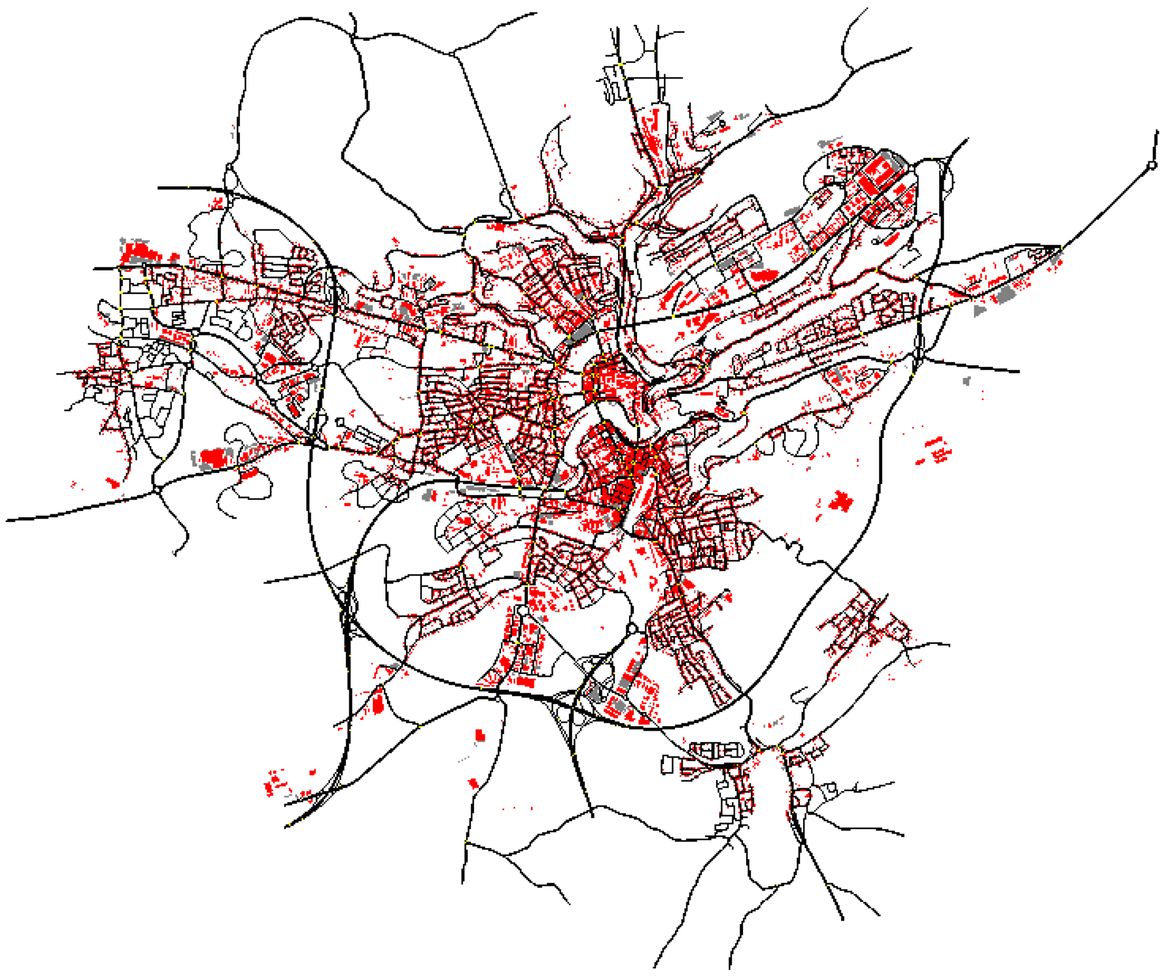


Figure 3.23: Luxembourg map that is used for the evaluation of RIoT

Figure 3.23 shows the Luxembourg map used in the simulation. For the best evaluation of RIoT, just realistic traffic patterns and volumes—which determine the mobility and number of vehicular bots in each individual area on the map—alone are not enough. We also need realistic locations of IoT devices based on their categories. Therefore, we designed a novel placement mechanism that identifies all the locations on a given map associated with each IoT device category. It first determines the keywords for each category using Google Maps data (e.g., industrial keywords are factory, manufacture, plant, etc.). After identifying these keywords, our mechanism queries Google Maps API for all the locations associated with each keyword. Finally, it builds a *location set* for each category that consists of the coordinates of all the locations related to the keywords for that category. For the evaluation of RIoT, we apply the placement mechanism to the Luxembourg map for the location search to be performed on; however, our mechanism can be used for the placement of IoT devices on any map.

3.4.2.2 RIoT Attack Mechanism

In the beginning of the attack, the vehicular bots sync their exploit database with the botmaster. The botmaster is expected to have the most up-to-date exploit database, and is responsible for adding new types of IoT devices and their vulnerabilities to its database as they become available, along with new vulnerabilities of the existing devices in the database. Once the vehicular bots obtain the most up-to-date exploit database from the botmaster, they start searching for any IoT device in their communication range as they move along their routes. The vehicular bots do not alter their original routes in any way for the purpose of finding more IoT devices, since it would otherwise be suspicious to the legitimate owners of these vehicles. For each IoT device detected in their communication range, they identify its type and query their exploit database for all applicable exploit payloads. Afterwards, they attempt delivering these payloads to the IoT device through a direct wireless ad hoc connection, bypassing any firewalls that might be installed on its gateway access point. This direct connection does not use GHOST—or any other VANET-based communication protocol—to deliver the exploit payloads; therefore, it is not subject to the limited bandwidth GHOST permits, which would adversely affect the payload upload speed. Due to the limited time for which the vehicular bots can hold the IoT device in their communication range, they deliver all applicable exploit payloads to it at once to maximize the probability of compromising it. Using the vehicular botnet communication, the vehicular bots share the areas that they covered so far with each other so that they can stay inactive in already covered areas in order to minimize the chance of getting detected.

RIoT is a highly effective mechanism for compromising the IoT devices in areas as large as a city, which would be infeasible for any single attacker. The mobility and collective communication range of vehicular bots, along with the variation in their routes, enable compromising a significant percentage of the IoT devices in a whole city in a very short amount of time. Also, since it is natural for vehicular bots to pass by IoT devices along their paths, compromising them in the background meanwhile would raise less suspicion, compared to a single attacker moving around to compromise them.

3.4.3 Evaluation

In order to fairly and accurately compare the effectiveness for different values of the attack parameters, we define the effectiveness metric as the percentage of IoT devices that the vehicular bots were able to compromise within a specified amount of time. The reason for setting a time limit is that the parameter values mostly affect the attack speed; due to the effectiveness of RIoT, given a sufficient amount of time, similar percentages of IoT devices get compromised regardless of these values. Therefore, we chose this time limit to be the first 6 hours of the simulation—after which the effectiveness of RIoT does not improve significantly with the optimal parameter values, and at which the effectiveness with different parameter values are still differentiable (see Figure 3.26). The effectiveness metric is calculated for each IoT device category separately in order to capture the differences in effectiveness between them. These differences are caused by the spatial characteristics of the devices under these categories.

We evaluated RIoT using rigorous and long experiments. We ran 60 simulations in total, each of which simulated the 24-hour Luxembourg scenario. For these experiments, we used two attack parameters that could alter the attack's effectiveness: vehicular bot percentage and communication range. Each graph in this section is produced by varying each relevant attack parameter while keeping the other one constant at its standard value—which is 20% for the vehicular bot percentage and 300 meters for the communication range (as defined by IEEE 802.11p [Wir10]). After each simulation, using the effectiveness metric described above, we calculated the effectiveness for all the IoT device categories. For each simulation, we introduced significant randomness in several components of RIoT so that conditions that could give high effectiveness were not constantly favored. First, implemented by the Luxembourg scenario, there is randomness in each vehicle's rerouting choice to load-balance traffic flows. Also, we determine which vehicle is going to be a vehicular bot randomly for each simulation. Finally, we place the IoT devices on the map by randomly selecting from the *location set* for each IoT device category—created by our placement mechanism—according to the percentages of each category (see Figure 3.22). Due to these random factors in each simulation, we ran 10 simulations for each pair of attack parameter values. The final effectiveness for each value on the x-axis of each graph are then calculated by averaging all the effectivenesses from 10 simulations.

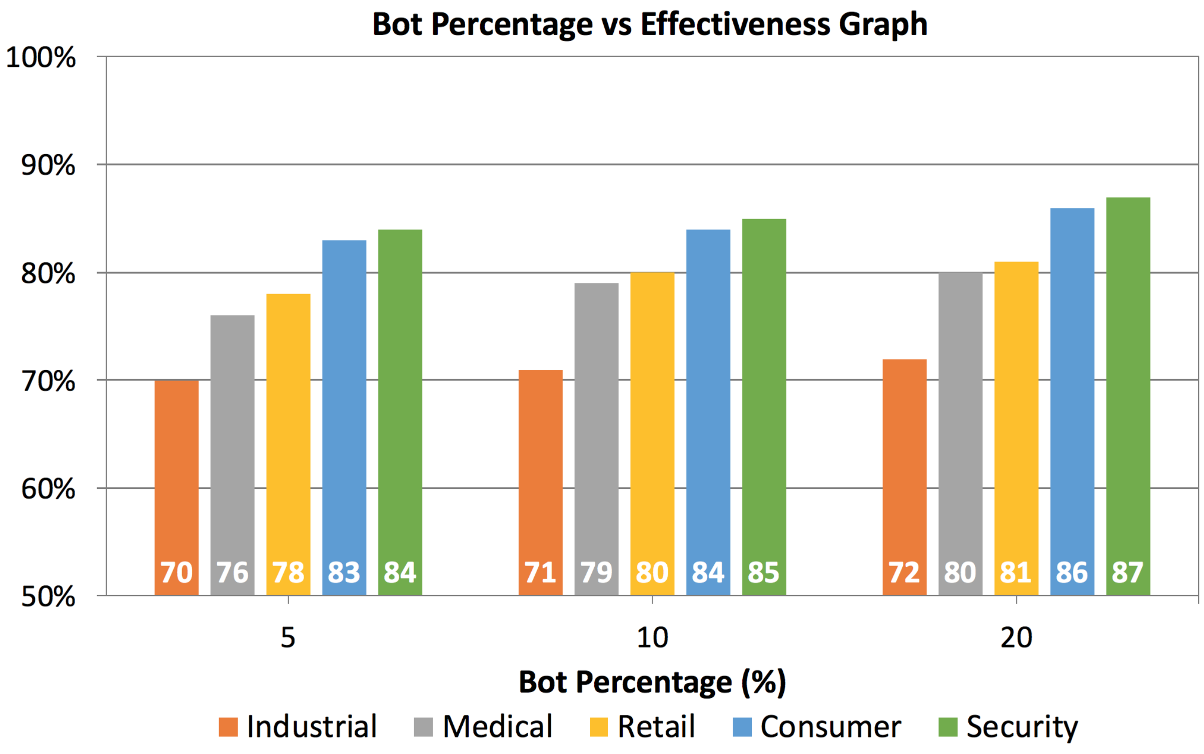


Figure 3.24: Effectiveness for each IoT device category with different percentages of vehicular bots

The effectiveness of RIoT is correlated with the size of the area on the map that is collectively covered by the vehicular bots. Figure 3.24 shows the effectiveness for all IoT device categories with each percentage of vehicular bots over the total number of cars created during simulation. The reason that the same category has similar effectiveness for different vehicular bot percentages is explained later in Figure 3.26. The most important factor that influences how the different values of the attack parameters affect the effectiveness of RIoT is the spatial characteristics of the IoT devices based on their categories. These characteristics are the geographical locations and types of the places where the devices under each category are expected to exist. For example, industrial IoT devices are mostly located in remote areas where the traffic is sparse, and industrial buildings such as factories are generally located in large campuses away from roads. Therefore, RIoT has the lowest effectiveness for this category since not many vehicular bots come in the vicinity regardless of the vehicular bot percentage due to the traffic sparsity, and if they do, their communication range is not sufficient to reach the devices in the large industrial campuses. Medical IoT devices are located in relatively less remote areas but still in large campuses such as hospitals. Retail IoT devices are in more urban areas and in smaller but still somewhat large campuses close to roads, such as shopping malls. Consumer IoT devices are in urban areas and located in buildings close to roads rather than campuses, whereas security IoT devices are found in any type of location.

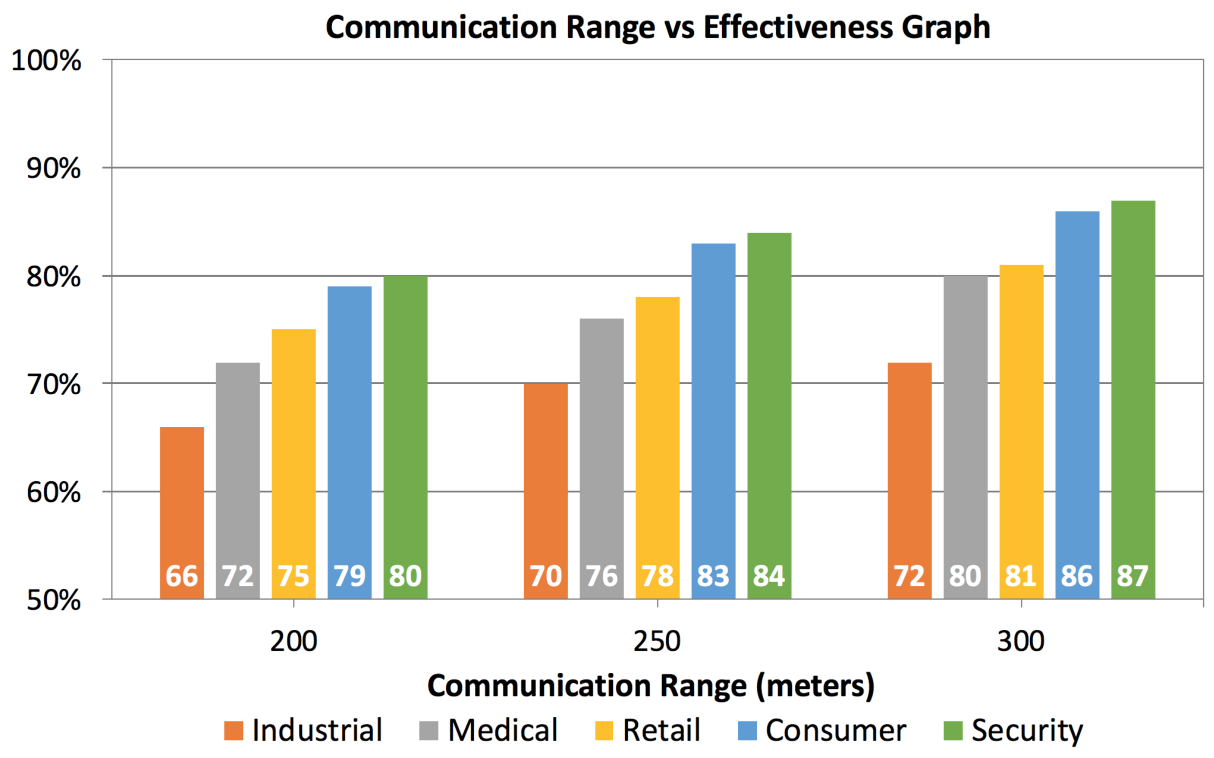


Figure 3.25: Effectiveness for each IoT device category with different wireless communication ranges

The effectiveness of RIoT is more sensitive to the communication range than the vehicular bot percentage. Figure 3.25 shows the effectiveness for all IoT device categories with each communication range. Since there is a correlation between the effectiveness and the size of the area covered by the vehicular bots, as stated earlier, the change trends in them are similar. For example, the covered area shrinks linearly with the number of vehicular bots while exponentially with the communication range due to being proportional to the communication range squared. As a result, the effectiveness of RIoT decreases exponentially with the communication range in Figure 3.25, whereas we observe a logarithmic decrease in the effectiveness with the vehicular bot percentage in Figure 3.24. In addition to the covered area size, there is also another reason why the communication range has a bigger effect on the effectiveness than the vehicular bot percentage. Although a smaller number of vehicular bots could be compensated for given an adequate amount of time, an insufficient communication range might have a more permanent impact on the effectiveness, as later discussed in Figure 3.26. With a limited communication range, some IoT devices might never be reachable regardless of the amount of time given or how high the number of vehicular bots is. Effectiveness especially for the industrial and medical IoT devices are even more affected by the decreases in the communication range, since they would have to be compromised over their long distances to roads.

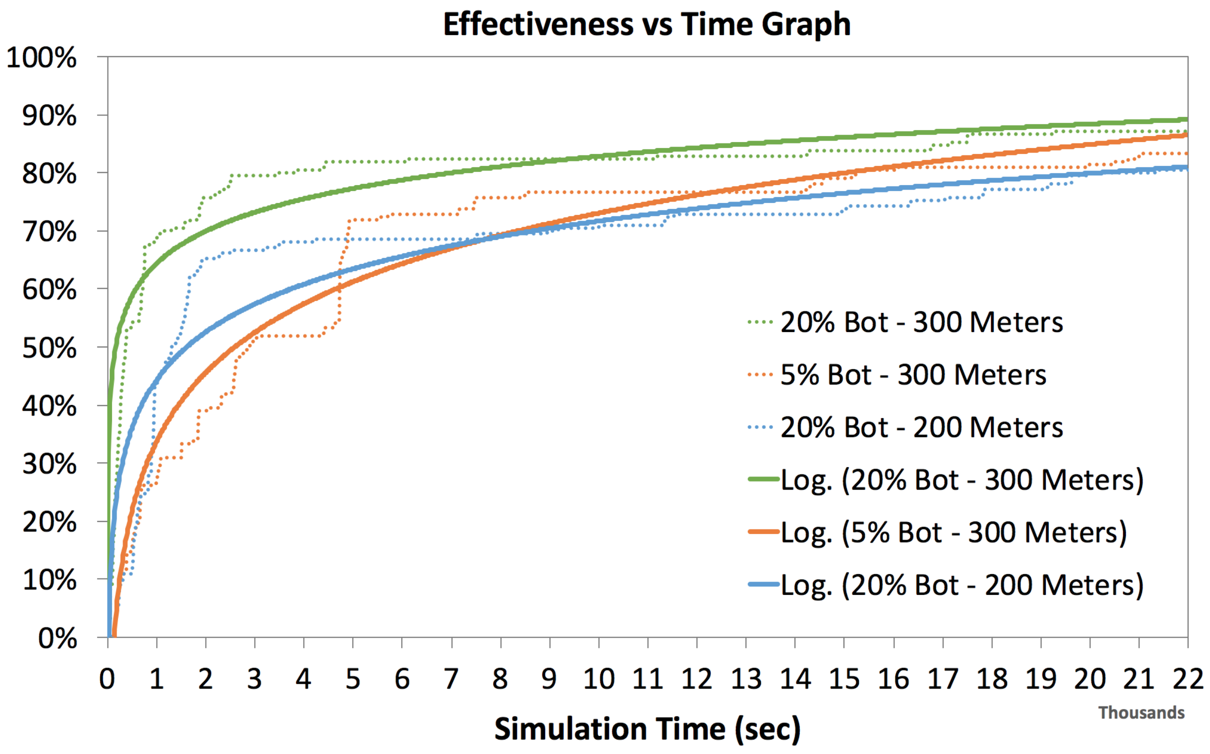


Figure 3.26: Attack speed of RIoT with different attack parameter values

Figure 3.26 shows the effectiveness of RIoT over time with different attack parameter values, and depicts how a low vehicular bot percentage impacts the attack speed, compared to a short communication range and vice versa. The attack speed is the slope of the effectiveness-time line. In Figure 3.26, values of the effectiveness over time are represented with a dotted line. They all follow a logarithmic trend line, which is represented with a solid line, as the attack speed decreases over time. It can be clearly seen in the graph that the vehicular bot percentage is the main factor in determining the initial attack speed, with the communication range still having a noticeable—though limited—impact. In both experiments with 20% vehicular bots, regardless of the communication range, the initial attack speed is much faster than in the experiment with 5% vehicular bots. However, between the experiments with 20% vehicular bots, the effectiveness grows faster initially in the experiment with a longer communication range, which demonstrates that the communication range still has some impact on the initial attack speed even though not as much as the vehicular bot percentage. Despite the initial attack speeds, the effectiveness-time lines almost flatten eventually, and we observe that the values of the attack parameters have different impact on the effectiveness of RIoT than they do on the initial attack speed. In this case, the communication range has a more significant impact than the vehicular bot percentage on the effectiveness in the long term. The effectiveness-time lines with the same communication range converge to similar values regardless of the vehicular bot percentage, which explains the similar accuracies for each IoT device category in Figure 3.24. This is also why we set the time limit for the effectiveness metric to the first 6 hours of the simulation; as shown in Figure 3.26, the effectiveness-time line with the optimal attack parameter values is almost completely flat after the first 6 hours, and each effectiveness-time line is still differentiable from the others. The communication range, on the other hand, has a more severe and permanent impact on the effectiveness. The effectiveness-time line with 20% vehicular bots and 200-meter communication range—despite a faster initial attack speed—falls and stays below the line with 5% vehicular bots and 300-meter communication range. Afterwards, it follows a logarithmic trend line, at lower accuracies, almost parallel to the line with 20% vehicular bots and 300-meter communication range.

3.4.4 Conclusion

In this work, we presented RIoT—a rapid exploit delivery mechanism that can compromise a significant percentage of the IoT devices in an area of interest as large as a city within a very short amount of time. In order to achieve this, RIoT takes advantage of the mobility and collective communication range of vehicular bots, as well as the diversity in their routes. It is the first attack in the literature against IoT devices using vehicles. For testing RIoT in conditions as realistic as possible, we first determined the percentage of each IoT device category and its expected number of devices. We then described a realistic 24-hour Luxembourg scenario that we used for the evaluation of RIoT, along with our novel mechanism that places the IoT devices in this scenario at realistic locations. We showed via realistic and thorough experimentation that RIoT can compromise up to 87% percent of the IoT devices in the whole city of Luxembourg within just the first 6 hours of the simulation. This work showed that vehicular botnets are threats not only to VANETs; they can also be used by attackers as an effective tool to perform powerful attacks against other important systems and networks.

3.5 In Closing

This chapter demonstrated that vehicular botnets could widen the attack surface of VANETs with novel, powerful and dangerous attacks by taking advantage of the collective power of vehicular bots. We described our three vehicular botnet attacks, each of which represented a different category of security threats posed by vehicular botnets—such as having physical consequences, violating privacy, and threatening other types of systems and networks than VANETs. The congestion attack showed that vehicular botnet attacks could have physical and possibly fatal consequences. BOTVEILLANCE demonstrated that they could also violate the privacy of VANETs. Finally, RIoT was an example of how vehicular bots could be used for attacks against other important systems and networks besides VANETs. Understanding the characteristics of these attacks is important for identifying possible countermeasures against them as a supplement to the all-inclusive defense against vehicular botnets discussed in the subsequent chapter.

CHAPTER 4

Mitigation Techniques

4.1 Introduction

Vehicular bots communicate with each other and coordinate their attacks by using GHOST, which is described in Chapter 2. While GHOST is not the only possible mechanism for vehicular botnets to coordinate their activities, future mechanisms are very likely to be similar to GHOST since using the VANET control channel for vehicular botnet communication is the stealthiest approach—given that the control channel is already standardized to be frequently used by everyone [Wir10]. Since only BSMs are allowed to be sent through this channel, future mechanisms will also be forced to work with BSMs to transmit botnet messages, which will have similar side effects to GHOST. Since vehicular bots have to use GHOST or a similar communication protocol to secretly coordinate their attacks, similar to Internet botnets, the most effective defense against such cooperative adversaries is targeting their communication protocol to identify them [NAP13]. Instead of trying to detect which specific vehicular botnet attack is being performed and defend against it, going after the common mechanism among such attacks—vehicular botnet communication—provides a defense against not only the known threats but also the future ones. In this work, we target GHOST since any detection mechanism against it can also largely be used against future mechanisms anyway due to the aforementioned reasons.

In this chapter, we present SHIELDNET, a detection mechanism against vehicular botnets, which applies machine learning techniques to search for evidence of GHOST usage and identify the participants as possible vehicular bots. Since the only indication of GHOST usage is the effects that its secret message injections have on the values of the injected BSM fields, each machine learning algorithm of SHIELDNET is chosen based on the expected change pattern in each corresponding field value to be able to detect the anomalies. Our work is the first implementation of a defense against vehicular botnets. After identifying the most applicable machine learning algorithms for the BSM fields, we determine their best configurations. However, SHIELDNET is designed in an adaptive manner, meaning that its machine learning algorithms can easily be replaced or extended without affecting its reputation-based vehicular bot identification component. We also discuss the vulnerabilities in the VANET standards in more depth and show via simulation that GHOST remains infeasible to detect even with SHIELDNET as long as these vulnerabilities exist. We then implement the vulnerability fixes so that our defense is evaluated with the improved standards. Finally, we demonstrate using a congestion attack scenario that, after SHIELDNET identifies the vehicular bots, their eviction from the network can significantly improve the severe conditions created by vehicular botnet attacks.

4.2 Related Work

The concept of botnets has received much research attention due to the high impact of these networks of compromised machines. [ZLC08] surveys the early research on the topic and [BCJ09] provides a taxonomy and classification of botnet technology and defenses.

Internet botnet detection, particularly, is a well-studied subject and there are numerous approaches proposed to tackle this problem. Most botnet detection techniques are based on passive network traffic monitoring and analysis [FSR09] [SCC09], similarly to our approach. These techniques attempt to detect botnets by identifying their C&C channels that are used for their coordination; some C&C protocol models are presented in [CSS10]. They try to achieve this by searching for anomalies in network traffic [GPZ08] [KRH07]—such as high latency, high traffic volumes, and traffic on unusual ports—or specifically in DNS traffic [BKK11], which are caused by C&C activities. However, the C&C channel of GHOST neither causes these network anomalies nor uses the same network architecture as Internet botnets. Vehicular botnet messages are already injected into BSMs, which are broadcast to everyone with a high frequency as per the VANET standards. This way, the C&C activities of vehicular botnets do not change the existing network conditions in any way. Therefore, Internet botnet detection approaches are not applicable to our problem.

In the context of VANETs, there is no existing work in the direction of detecting vehicular botnets. Our work is the first step towards identifying the unique characteristics of vehicular botnets to detect them, and the first implementation of such a detection mechanism. We determine the standard and most applicable machine learning algorithms [Cho17] [EKS96] [Kaf17] based on the expected change patterns in the injected BSM field values and the format of each field, and use these algorithms to detect anomalies in them. The scope of this work is not designing new machine learning algorithms; our novelty comes from designing an adaptive vehicular bot detection framework that houses them. The framework provides organized and efficient data collection for the built-in machine learning algorithms, and uses their outputs for its reputation-based vehicular bot identification mechanism. Its adaptive design makes the addition of new machine learning algorithms or the removal of the built-in ones very easy for future researchers.

The vehicular bots that are identified by our detection mechanism would naturally need to be removed from the network. Since there are already many existing mechanisms for the eviction of misbehaving vehicles—such as [MRC08] and [RPA07]—which are surveyed in [RBZ11], designing a new one is out of the scope of this work. Instead, we show the effect of using one to evict the vehicular bots in Section 4.6.

4.3 SHIELDNET

Here we describe the design details of SHIELDNET—the machine learning algorithms applied on the injected BSM fields to detect anomalies in their values and the reputation-based analysis of these anomalies to identify vehicular bots. SHIELDNET is intended for use by authorities with the means to monitor the exchanged BSMs in the whole *shielded area*. It is designed in an adaptive manner so that authorities can easily replace the machine learning algorithms applied on the data whenever they feel necessary without affecting the reputation-based identification of vehicular bots. We use a labeled dataset, which is a large set of BSMs—collected during our vehicular botnet simulation—where the injected ones are marked, for the investigations and evaluation in this work; however, SHIELDNET uses unsupervised machine learning algorithms and does not require any labeled data to identify vehicular bots.

These algorithms detect the anomalies in the injected BSM fields by predicting their values based on the features and learned behavior specific to them. Since there is no predictable behavior for the positional accuracy field—the noise in the calculation of the vehicle's heading relative to true north—only the speed, latitude and longitude fields are used for the anomaly detection.

4.3.1 Anomaly Detection in Speed

Prior to applying machine learning to detect anomalies in speed values, we performed statistical analysis on the labeled dataset to look for a predictable pattern in how the GHOST injections might effect these values. We first investigated if the injections yield higher or lower speed values on average than the non-injected values. Since the effect of the four-bit injections on the speed values in two-byte granularity is very low as aforementioned, we calculated the average of values represented by the four least significant bits of the speed field for all the injected and non-injected BSMs. We observed that these two averages were nearly identical, meaning that there was no predictable pattern. We repeated the same analysis for the one, two and three least significant bits as well but the outcome was the same. We then investigated if there was a bit pattern associated with the injections in the speed values, that is, if the injections cause more ones or more zeros in the binary speed values on average than the non-injected values. We calculated the average number of ones and the average number of zeros in the speed field for all the injected and non-injected BSMs. Since the content of the injection is random due to the botnet message being encrypted with a random scheme that changes periodically, this analysis did not reveal any predictable pattern as well. Due to the unpredictability of the effects of the injections caused by the randomness in their nature, the only effective approach to detect anomalies in speed values is learning a vehicle's recent mobility pattern and determining the plausibility of its following speed broadcasts based on how well they fit this pattern.

SHIELDNET uses a machine learning algorithm, specifically moving average using *discrete linear convolution* [Cho17], to model each vehicle's mobility pattern based on its speed advertisements and detects outliers with a stationary standard deviation. Each outlier is then reported to the reputation-based vehicular bot identification mechanism.

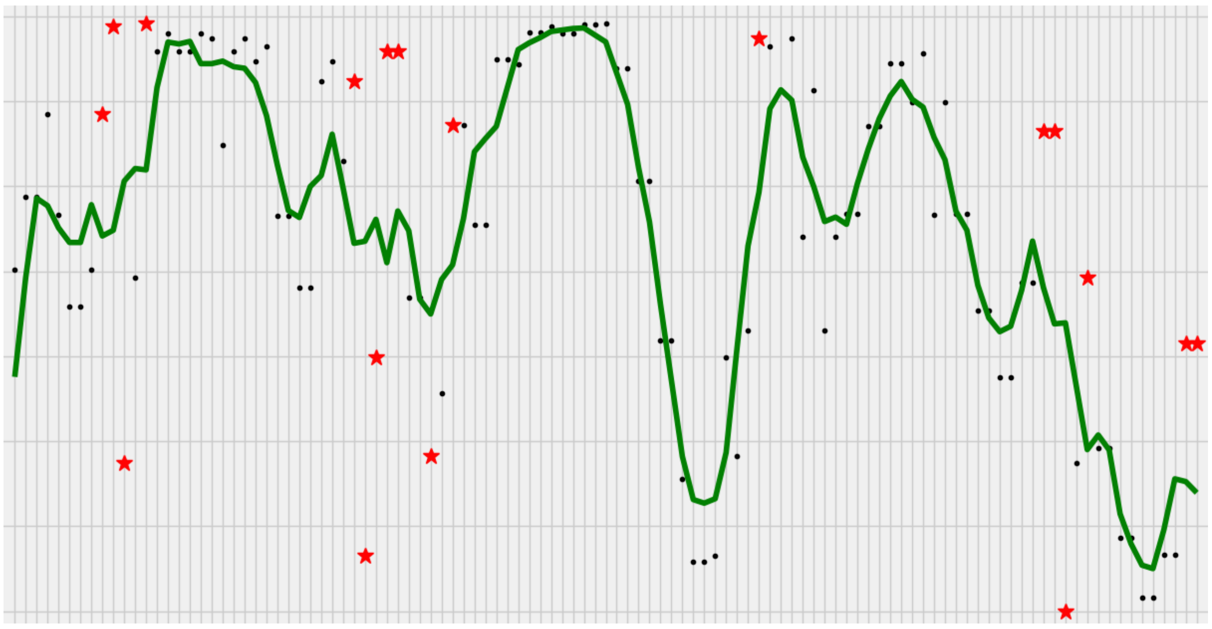


Figure 4.1: Model of a vehicle's mobility and outliers during the simulation

SHIELDNET creates a separate mobility model for each vehicle and performs anomaly detection simultaneously for all of them. Figure 4.1 shows the mobility model created for one of the vehicles in the simulation (green line). The model is continuously updated with each speed advertisement from the vehicle received by authorities monitoring the BSM broadcasts in the *shielded area*. Each advertised speed value (black dot) is tested against the model built so far and marked as an outlier (red star) if it does not fit the model based on the configurable stationary standard deviation. If it is not considered as an outlier, the model is then updated to include this new speed value in its convolution.

4.3.2 Anomaly Detection in Latitude and Longitude

Detecting anomalies in latitude and longitude values by learning their change patterns is impractical for several reasons. First, depending on the heading, only the latitude or only the longitude or both might change and the amount of change in each is determined by both the heading and speed. As a result, the convolution approach alone would not be able to predict their next values. Second, even if another machine learning approach could incorporate the effect of heading and speed in its model, there is another factor in addition to the effect of injections that might be considered as an anomaly: GPS error. Therefore, the only effective approach to detect anomalies in latitude and longitude values is learning the GPS error pattern to be able to distinguish the effect of the injections from the GPS error as anomalies.

The level of GPS error is determined by the environmental factors such as interference, obstacles, etc. Therefore, vehicles under the same environmental factors are expected to have similar GPS errors. In other words, a vehicle having a significantly different GPS error than other vehicles under the same environmental factors would indicate an anomaly—likely to be caused by the injections into the vehicle's latitude and longitude values. In order to detect such anomalies, GPS errors in the same environment need to be clustered together based on their similarity. Since being in the same environment is represented as having both similar latitudes and longitudes, 2D clustering approaches would fail to model one of these dimensions required: latitude, longitude and GPS error.

SHIELDNET uses the 3D version [Kaf17] of DBSCAN [EKS96]—a density-based cluster discovery algorithm for large spatial databases with noise—for dividing the spatial GPS errors into multiple 3D clusters based on their similarity and proximity.

SHIELDNET's GPS error calculations start for each vehicle when its first two BSMs, say BSM_1 and BSM_2, are received. Using the speeds and headings advertised in BSM_1 and BSM_2, as well as BSM_1's and BSM_2's timestamps, BSM_2's expected latitude and longitude—lat_{exp} and $long_{exp}$—are calculated. The GPS error for the vehicle would then be the Euclidian distance between lat_{exp} and $long_{exp}$ and the actual latitude and longitude advertised in BSM_2—lat_{adv} and $long_{adv}$. The next GPS error for BSM_2 and BSM_3 is calculated using BSM_2's lat_{exp} and $long_{exp}$ instead of its lat_{adv} and $long_{adv}$, which are inaccurate due to the GPS error.

Figure 4.2 shows how the calculated GPS errors are divided into multiple spherical 3D clusters (different colored circles). Clusters are created based on the configurable maximum cluster radius and minimum number of points required within this radius to form a cluster. Figure 4.2 depicts only the data belonging to a subset of the *shielded area*. The latitude (y-axis) and longitude (x-axis) values in the graph are the distance in meters from the bottom and left edge of the *shielded area*, respectively. Each calculated GPS error is tested against the model built so far and marked as an outlier (blue cross)

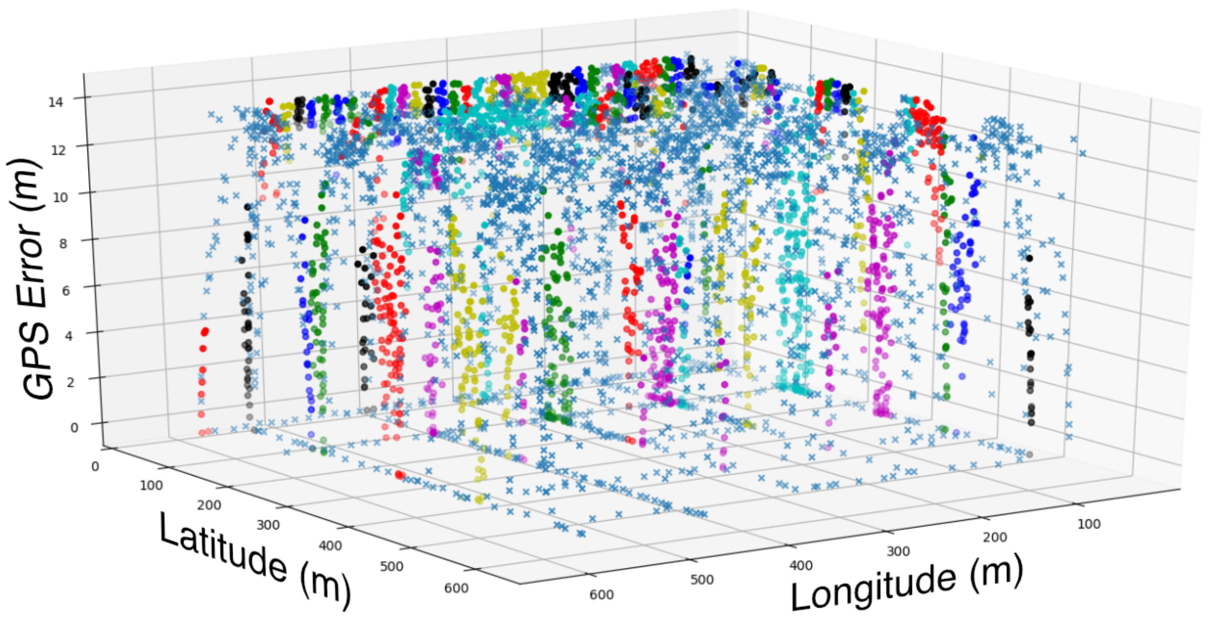


Figure 4.2: 3D clusters of the GPS errors and outliers calculated by DBSCAN

if it cannot belong to a cluster. The outlier is then reported to the reputation-based vehicular bot identification mechanism.

4.3.3 Reputation-Based Identification of Vehicular Bots

SHIELDNET performs a reputation-based analysis on all of the outliers reported by the built-in machine learning algorithms in order to identify vehicular bots. This identification mechanism is designed to provide authorities with a list of vehicles that are most likely to be bots and to constantly update this list while the machine learning algorithms simultaneously generate new outlier reports.

Figure 4.3 shows the individual components of SHIELDNET, how they communicate with each other and the input/output of each component. These components operate in a pipelined manner; each component immediately processes the partial data inputted and forwards the result to the next component while the new data is constantly being inputted. This pipeline starts with the *data collector* where all the BSMs being broadcast in the *shielded area* are inputted. Authorities collect this data by using network sniffers with a high communication range—most likely the RSUs. The *data collector* then organizes the inputted BSMs and passes them to the machine learning algorithms as input. There can be any number of machine learning algorithms at the next step in the pipeline and it

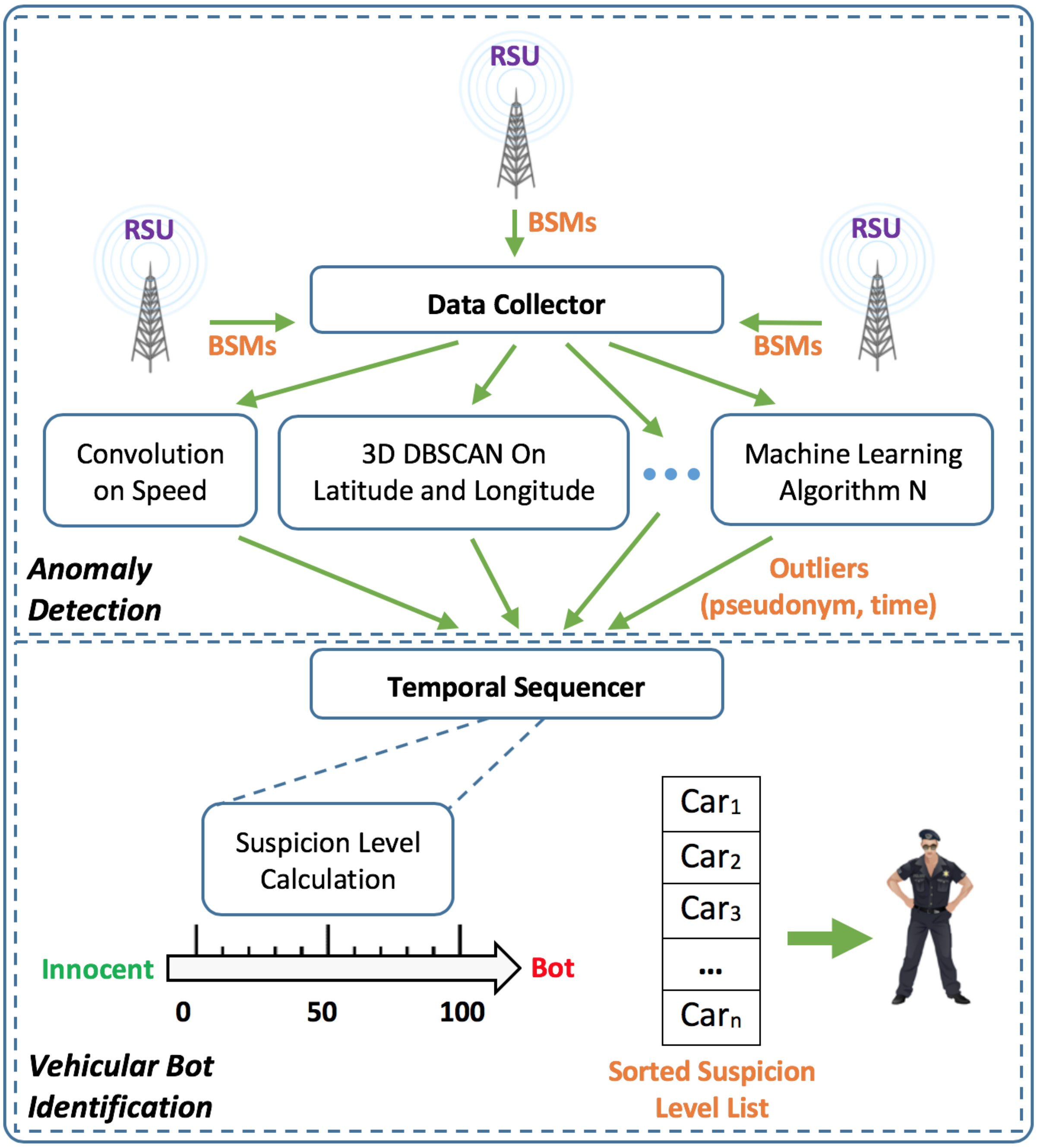

RSU
BSMs
RSU
RSU
Data Collector
BSMs
BSMs
Convolution on Speed
3D DBSCAN On Latitude and Longitude
• • •
Machine Learning Algorithm N
Anomaly Detection
Outliers (pseudonym, time)
Temporal Sequencer
Suspicion Level Calculation
Innocent
Bot
0
50
100
Car1
Car2
Car3
...
Carn
Sorted Suspicion Level List
Vehicular Bot Identification

Figure 4.3: Diagram of the SHIELDNET's components and their interactions

might change over time. Therefore, the *data collector* is designed in an adaptive manner to detect the current algorithms and forward the data accordingly. After BSMs are passed to each machine learning algorithm, outlier detection is performed on the data relevant to the algorithm and outliers are reported to the *temporal sequencer*. Each outlier report is a pair of the pseudonym of the vehicle that broadcasts the outlier and the timestamp of this broadcast. Based on these reports, the *temporal sequencer* calculates the suspicion level for each vehicle, which determines its likelihood of being a bot. Finally, the *temporal* *sequencer* sorts the list of vehicles by their suspicion levels and marks all the vehicles with suspicion levels that are higher than a threshold as bots. The pseudonyms of these vehicles that are likely to be bots are then given to authorities for the appropriate action to be taken towards either recovering their system to its factory settings or removing them from the network.

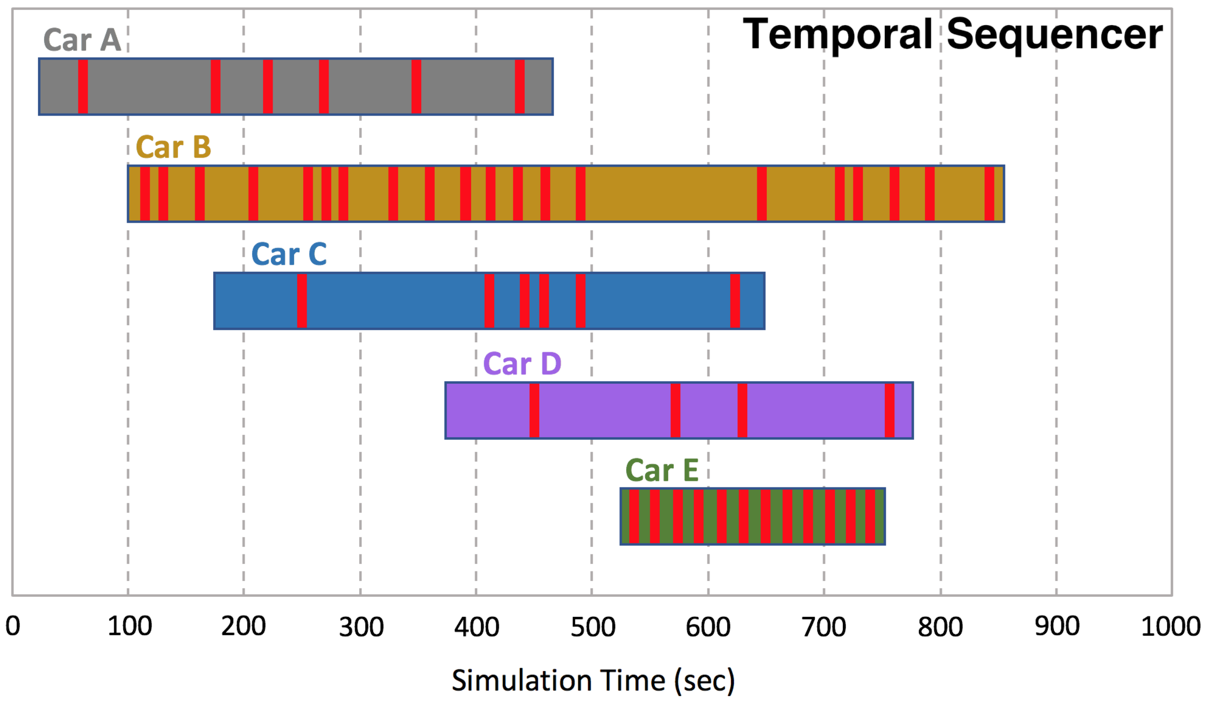


Figure 4.4: Temporal sequencing of the outliers to calculate suspicion levels

The mechanism for the temporal sequencing of the outliers reported by the machine learning algorithms is depicted with an example in Figure 4.4. Starting from the first BSM, every broadcast of each vehicle in the *shielded area* is placed in the *temporal sequencer* ordered by its timestamp (different colored bars), which captures both the duration that each vehicle is in the *shielded area* relative to others and the total number of BSMs received from each vehicle. The BSMs with an outlier reported by any of the machine learning algorithms are marked by the *temporal sequencer* (red lines) according to the pseudonyms and timestamps located in the outlier reports. These marked BSMs are then used for calculating the outlier percentage of each vehicle—which is the percentage of the marked BSMs of a vehicle over all of its broadcast BSMs—to be used in the suspicion level calculation.

Outliers might occur for every vehicle due to the false positives caused by the machine learning algorithms. As a result, an innocent vehicle might sometimes have the same outlier percentage as a vehicular bot. However, due to the nature of GHOST and vehicular botnet attacks, injections and the outliers caused by them are likely to be frequent during certain times whereas the outliers for innocent vehicles tend to be more scattered. For example, even though Car A and Car C in Figure 4.4 seem to have the same outlier percentage, Car C is more likely to be a bot due to this reason. Therefore, the *temporal sequencer* divides the BSM and outlier report data into multiple time intervals (dashed lines) and performs the outlier percentage calculations for each interval separately. If a vehicle's outlier percentage for an interval is above a threshold, the vehicle gets marked as a violator for that interval. The *temporal sequencer* then calculates the suspicion level for a vehicle by finding the percentage of the intervals for which the vehicle is a violator over all the intervals that the vehicle's BSMs span. The *temporal sequencer*'s use of percentages—outlier percentage of a vehicle instead of its total number of outliers and percentage of a vehicle's interval violations instead of the total number of interval violations—normalizes the effect of vehicles being in the *shielded area* for different durations on the resulting suspicion levels. For example, in Figure 4.4, it enables Car E to be correctly assigned a higher suspicion level than Car B.

After the calculation of the suspicion level for a vehicle, it is immediately added to the *temporal sequencer*'s sorted suspicion level list and authorities are updated with the new list. If the suspicion level of the vehicle is above a threshold, it gets identified as a bot and authorities deal with it alongside the other identified bots in the list. Note that none of the outlier percentage calculation, suspicion level calculation or vehicular bot identification mechanism depends on any specific types or number of machine learning algorithms. This is because, much like the *data collector*, the *temporal sequencer* is also designed in an adaptive manner so that the types and number of the machine learning algorithms used for anomaly detection can easily be changed as long as there is at least one algorithm that outputs outlier reports.

4.4 Evaluation

Since the vulnerability in the VANET standards that makes GHOST infeasible to detect is the excessive fine granularity in the injected fields, we evaluated SHIELDNET using both the BSM traces with the normal granularity that is compliant with the current standards and the traces with the decreased granularity. We decreased the granularity of the injected fields by a magnitude that makes the effect of the injections more noticeable and that we believe does not affect traffic safety; the details are explained in Section 4.5. Each simulation was 30 minutes long and a total of 1500 cars passed through the *shielded area*. The percentage of vehicular bots over the total number of cars was 20%.

We evaluated SHIELDNET separately for each machine learning algorithm, that is, there was only one active machine learning algorithm in each simulation detecting anomalies in the values of the specific injected BSM field(s) it is responsible for. This way, as we observed the capabilities provided by SHIELDNET, we also evaluated the effectiveness of each machine learning algorithm individually. Each algorithm was tested with both the normal and decreased granularity BSM traces. The accuracy metric is defined as the percentage of vehicles that are actually bots in the list of identified vehicular bots SHIELDNET reports to authorities.

Convolution	Standard Deviation	Sliding Window Size	Interval Length (sec)
Normal Granularity	0.1	25	140
Decreased Granularity	0.5	5	40
3D DBSCAN	Max Cluster Radius (m)	Min Number of Points	Interval Length (sec)
Both Granularities	4	8	100

Figure 4.5: Optimal configuration parameters of the machine learning algorithms for the normal and decreased granularity

Figure 4.5 shows the optimal parameter values used in the experiments for the components of SHIELDNET—standard deviation and sliding window size for the convolution, maximum allowed radius for the clusters and minimum number of data points required to form a cluster for the 3D DBSCAN, and interval length used in the *temporal sequencer*. The factors that affect the optimal parameter values are different for each component. The optimal configuration of the convolution is determined by the used granularity since the sensitivity required to detect the outliers caused by injections changes with the granularity. On the other hand, the optimal configuration of the 3D DBSCAN is independent of the granularity but rather governed by the geography and its environmental factors that affect GPS errors and the best way to model them. Finally, the optimal interval length for the *temporal sequencer* is decided based on the frequency of the outlier reports from the machine learning algorithms; the more frequently outliers are reported, the smaller the interval length should be.

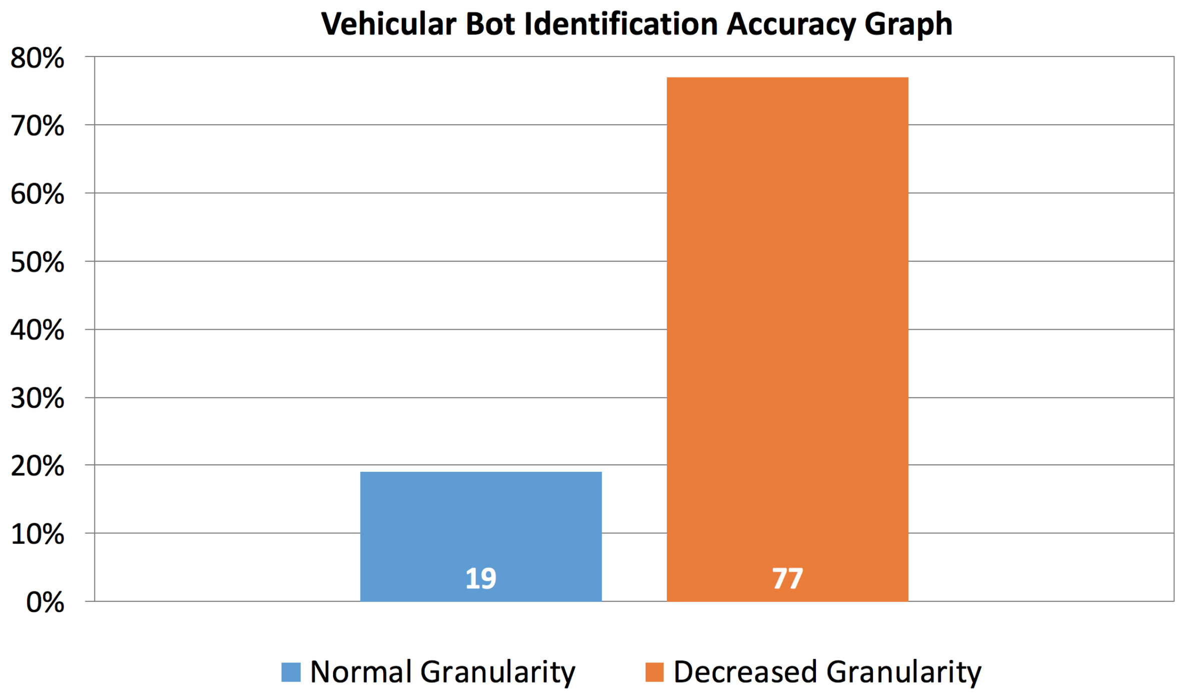


Figure 4.6: Accuracies of vehicular bot identification using convolution for the normal and decreased granularity

The vehicular bot identification accuracies are shown in Figure 4.6 for the normal and decreased granularity when only the convolution is used for anomaly detection. The results confirm that, with the normal granularity in the speed field as per the current VANET standards, GHOST remains infeasible to detect since 19% accuracy is just a statistical result of 20% of all cars in the simulation being bots. When the granularity is decreased, the experiments reveal that, with our approach, the time-series speed data indeed follows a sufficiently predictable pattern to detect injections and SHIELDNET can identify the vehicular bots with 77% accuracy.

Figure 4.7 shows the vehicular bot identification accuracies for the normal and de-

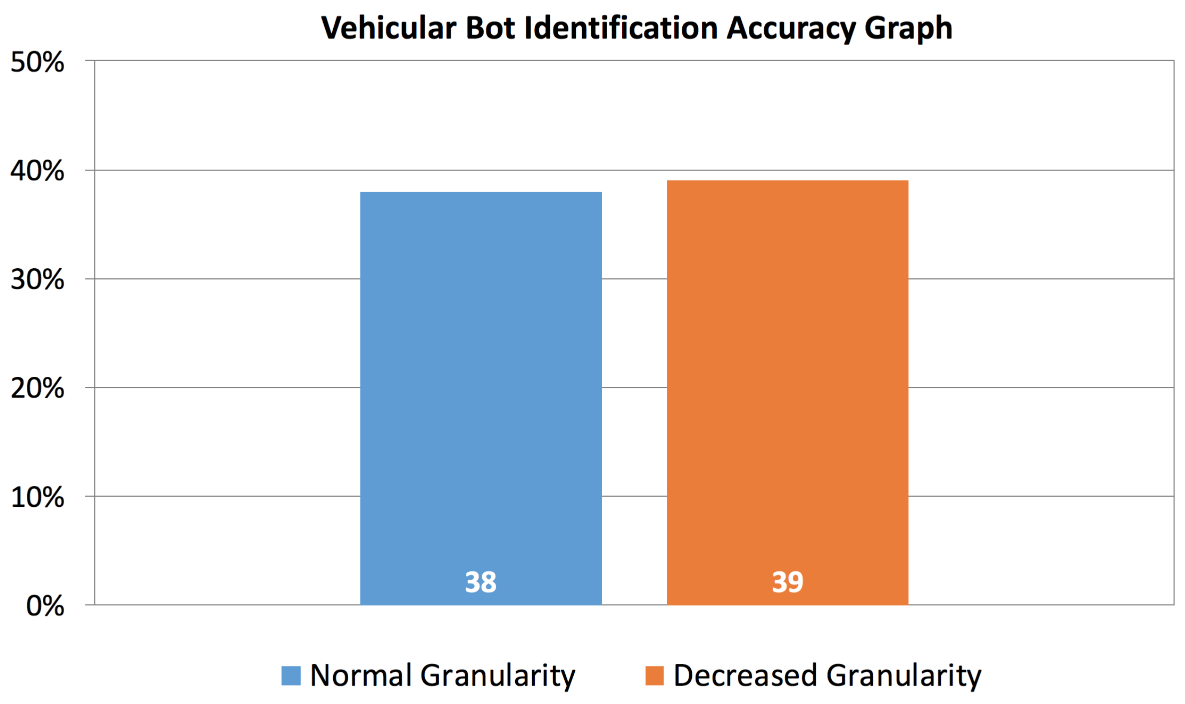


Figure 4.7: Accuracies of vehicular bot identification using 3D DBSCAN for the normal and decreased granularity

creased granularity when only the 3D DBSCAN is used for anomaly detection. The results substantiate the effectiveness of our approach of clustering GPS errors in 3D based on latitude and longitude; despite the normal granularity, 38% identification accuracy is significantly more than a statistical result of the bot percentage of 20%. On the other hand, decreasing granularity does not improve the accuracy that much due to the low limit of granularity decrease in the latitude and longitude fields that is possible without affecting traffic safety. This issue and an existing vulnerability related to the nature of GPS errors—which adversely affect the accuracy—are explained in Section 4.5.

The accuracy of SHIELDNET is also affected by the values of thresholds used in the *temporal sequencer*—the outlier percentage threshold to determine interval violators and the suspicion level threshold to identify vehicular bots. We currently use a static outlier percentage threshold, which is the same for all intervals, and a static suspicion level threshold. As future work, implementing a mechanism that determines the best threshold values dynamically based on changing network activity will significantly improve the SHIELDNET's vehicular bot identification accuracy.

4.5 Vulnerabilities in VANET Standards

The significant contributions of SHIELDNET to VANET security need to be complemented with improvements to the VANET standards since GHOST cannot be detected by SHIELDNET with the current fine granularity in the injected fields—each increment is in the unit of 0.02 m/s ($\approx$ 0.04 miles/h) for speed and 10^{-7} degrees for latitude and longitude [Wir10], which are the fields that the anomaly detection is performed on. Therefore, we decreased the granularity of these fields by a factor of 16, which we believe is the minimum granularity that does not affect traffic safety; a further decrease would cause each increment in the speed field to be in the unit of more than $\approx$ 1 mile/h, and each increment in the latitude and longitude fields to represent more than $\approx$ 1 meter in distance. The experiments showed that, while this granularity decrease by a factor of 16 was sufficient to make the injections on the speed field noticeable, the variations in GPS errors were still generally higher than the effects of the injections on the latitude and longitude fields. In other words, GHOST could not be detected without a significant false positive rate by using only the latitude and longitude values. Therefore, the standards should further be improved so that the additional granularity decrease in the latitude and longitude fields would not impact traffic safety (e.g., enforcing an extra safety distance). An obvious attacker response to the granularity decrease in the injected fields would be decreasing the size of the botnet messages to make injections less noticeable. However, this would significantly impact the functionality of GHOST due to the consequent increase in duration for fully synchronizing the information needed for attacks among vehicular bots, and decrease in the number of attacks that could be performed simultaneously, details of which are described in Section 2.3.1.2.

Another existing vulnerability that affects the success of the anomaly detection in latitude and longitude values is the unpredictability of GPS errors. There are several factors determining these errors such as satellite orbits, inaccuracies in satellite time, atmospheric effects, signal blockage, multi-path effect and radio interference [Jam17]. Randomness in these factors makes the GPS error for an individual vehicle unpredictable at any given time [JGO06]. However, since vehicles close to each other would be subjected to similar environmental conditions, they would have similar GPS errors. Our 3D DBSCAN model exploits this phenomenon effectively to detect the anomalies that are caused by the injections, as aforementioned. While our approach compensates for the randomness in GPS errors considerably, there is still a significant unpredictability in radio interference levels. Even vehicles close to each other might sometimes experience very different radio interference levels due to the changing number, positions and usage pattern of nearby interfering devices, causing a high false positive rate in the 3D DBSCAN. Therefore, as future work, techniques for smoothing random GPS errors [JGO06], prediction models for local radio interference caused by user devices, and/or methods for normalizing the effects of radio interference locally can be built and incorporated into our 3D DBSCAN model in order to decrease the differences in GPS errors of vehicles close to each other. Otherwise, even though SHIELDNET already has an identification accuracy of 77% using only the convolution on speed values, attackers might come up with ways to better utilize the latitude and longitude fields for injection, reducing the use of the speed field and thus decreasing the probability of getting detected.

4.6 Eviction of Vehicular Bots

The identification of vehicular bots needs to be followed by their eviction. After SHIELDNET provides authorities with the list of identified vehicular bots, they will be removed from the network by using an eviction mechanism—they will be unable to join again until their system is recovered to its factory settings. As discussed in Section 4.2, there are already many existing eviction mechanisms that can be used for removing the vehicular bots. Therefore, rather than designing a new one, we show the effect of the vehicular bots' eviction in a scenario where our congestion attack is being performed.

As described in Section 3.2, the congestion attack decreases the average speed on any chosen road by causing severe congestion on it. Figure 4.8 shows the average speed on the targeted road in the scenario where the attack starts at T=0. Since the congestion reaches severe levels around 300 seconds after the beginning of the attack, we simulate the complete eviction of the vehicular bots at that time to evaluate its effect in the harshest conditions. Despite the slight decrease in the average speed for a little longer after the eviction due to the residual impact of the already disseminated malicious information, the

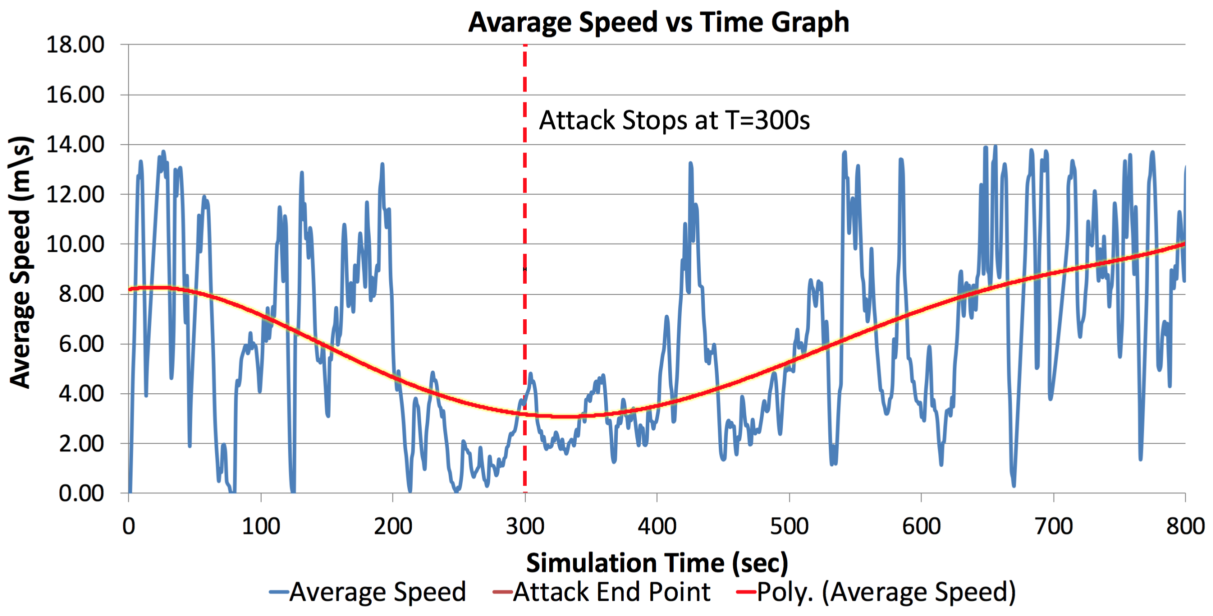


Figure 4.8: Effect of evicting vehicular bots 300 seconds after the congestion attack starts

congestion levels start improving significantly. The graph follows a continuous increase on the polynomial trend line, drawn in Figure 4.8, until it reaches the normal average speed values before the congestion occurred.

4.7 In Closing

In this chapter, we presented SHIELDNET—an adaptive detection mechanism designed to defend against vehicular botnets. It is a vehicular bot identification framework that employs a set of machine learning algorithms to detect the use of GHOST, a vehicular botnet communication protocol. SHIELDNET is the first implementation of a defense mechanism against vehicular botnets. We used the standard machine learning algorithms that are the most suitable for our anomaly detection approaches to evaluate SHIELDNET. We discussed the effectiveness of these approaches and explained how easily our adaptive framework can be extended with new machine learning algorithms. We showed via experimentation that GHOST remains infeasible to detect if the standard granularity in the injected BSM fields is not decreased. We described how we decreased the granularity of these fields without affecting traffic safety and demonstrated that SHIELDNET can identify 77 percent of the vehicular bots with the decreased granularity. We discussed the vulnerabilities that still exist in the VANET standards and need to be fixed. We recommended several solutions to improve the accuracy of the anomaly detection in GPS errors as future work. Finally, we demonstrated the effect of evicting the vehicular bots after SHIELDNET identifies them and how their eviction can significantly improve the severe conditions created by vehicular botnet attacks, by using a congestion attack scenario. Further future work on the general defense against vehicular botnets is discussed in the subsequent chapter, along with the possible countermeasures against individual vehicular botnet attacks as a supplement.

CHAPTER 5

Conclusion and Future Work

This dissertation introduced the concept of a vehicular botnet—a collection of cooperative adversaries in VANETs. Vehicular botnets are powerful entities due to the collective capabilities of their members and to the inability of existing VANET security solutions to cope with them. They widen the attack surface of VANETs with threats that have not been considered by earlier research. Therefore, it is of vital importance to thoroughly investigate their capabilities since the assessment of all possible sorts of threats against VANETs is required for the deployment of these networks. This dissertation is the first that ever proposed and designed vehicular botnets in the literature, thus also the first that could study and defend against them. The contributions of this work can be summarized as follows:

***Design of Vehicular Botnets*:** We first stated that attackers can compromise individual vehicles by using vulnerabilities existing in their systems and that there are already several such attacks presented in earlier work. We then argued that, similar to Internet botnets, these compromised cars can be organized into vehicular botnets. We showed how they can be organized using the VANET infrastructure and presented our VANET-based botnet communication protocol, GHOST, that hides itself in the VANET control channel so that the vehicular bots can coordinate their attacks without getting detected. We identified the vulnerabilities in the VANET standards that allow our vehicular botnet communication to remain hidden. We discussed existing VANET security solutions, the threats they aim to defend against, and their assumptions about the behavior of malicious vehicles. We argued that these assumptions are not accurate in the case of vehicular botnets, and thus existing VANET security solutions cannot mitigate them. We asserted that the most effective defense against them, as all other types of botnets, is investigating the characteristics of the communication they use to perform their attacks and designing defense mechanisms to disrupt it.

***Vehicular Botnet Attacks*:** In order to demonstrate the capabilities of vehicular botnets and how they can widen the attack surface of VANETs, we presented three vehicular botnet attacks, each of which represents a different category of security threats posed by vehicular botnets. We described our congestion attack, which can cause severe congestion levels on any chosen road. In a short amount of time, the attack not only makes the targeted road virtually unusable, but also the roads surrounding it. The congestion attack demonstrated that vehicular botnet attacks can have physical and possibly fatal consequences. We then presented BOTVEILLANCE—an adaptive cooperative long-range global-scale tracking attack—along with its novel built-in localization mechanism, INTERLOC. We provided a comprehensive survey on existing location privacy solutions for pseudonymous systems in VANETs, with discussions of their weaknesses. BOTVEILLANCE showed that, besides their physical consequences, vehicular botnet attacks can also have a damaging effect on location privacy, which is one of the most important VANET security requirements. Finally, we discussed RIoT, which is a reactive vehicular botnet attack targeting IoT devices in any area of interest where the attacker takes advantage of the mobility of vehicular bots to compromise as many IoT devices as possible along their routes. RIoT was an example of the capability of vehicular bots to serve as an effective tool for powerful attacks not only against VANETs, but also other important systems and networks. The investigation of these attacks paved the way for a better understanding of the vehicular botnet characteristics and provided guidance on how to defend against them.

***Mitigation of Vehicular Botnets*:** Since the most effective way to defend against any type of botnet is targeting its communication protocol, we presented a general defense mechanism against vehicular botnets—SHIELDNET—which is a vehicular bot identification framework using machine learning algorithms to detect the use of GHOST. We argued that our framework is highly adaptive and easily extensible with new machine learning algorithms. We described the vulnerability in the VANET standards—unnecessary fine granularity in the injected BSM fields—that we fixed to make GHOST detectable. We recommended further solutions to improve the effectiveness of SHIELDNET as future work. Finally, by using a congestion attack scenario, we demonstrated the effect that the vehicular bots' eviction after SHIELDNET's identification could have on undoing the damage caused by vehicular botnet attacks.

The pioneering work on vehicular botnets presented in this dissertation is a significant step towards understanding the characteristics of these powerful entities and defending against them. The wider attack surface provided enables more comprehensive research on the security and privacy of VANETs, which is required for their deployment. Future work to this dissertation—such as the possible countermeasures against individual vehicular botnet attacks in addition to and in support of SHIELDNET, and another supplement to the general defense against vehicular botnets—is discussed in the subsequent sections.

5.1 Congestion Attack

A reputation-based mechanism may be useful in detecting and isolating vehicular bots advertising malicious congestion measurements since each measurement can be traced back to its creator. Vehicular bots may advertise congestion levels that are inconsistent with each other in the big picture; therefore, an anomaly detection mechanism can analyze congestion advertisements for inconsistencies. In addition, other unusual aspects of the malicious congestion measurements might also serve as clues of the attack. For example, it would be suspicious for a car to advertise a very fresh measurement while the car is very far from the measured road, provided that the car does not lie about its current location too. Similarly, a car's measurements for roads far from each other would not normally have similar freshness, given the trip times required to travel between them. In addition, it would also be suspicious if a car seems to have measurements for too many roads that the car is unlikely to have visited.

The countermeasures mentioned above address only the problem of identifying the vehicular bots after the attack. It is also important to detect the onset of such an attack before it causes the damage. In the attack directed to the V2V congestion avoidance scheme, a possible remedy involves each vehicle simultaneously monitoring congestion levels from both V2V and cloud-based congestion avoidance mechanisms. As discussed earlier, the cloud-based applications like Google Maps generally update their knowledge of the local congestion levels slower than the V2V congestion avoidance mechanisms; on the other hand, their congestion reports are advertised directly over the Internet, which the vehicular bots are not able to interfere with. Also, these reports are timestamped and include the positions of the roads they belong to. Therefore, they can be used as (delayed) ground truth to correlate with the congestion measurements already disseminated through V2V advertisements. In other words, while the cloud-based traffic applications do not react as quickly to traffic jams as the V2V ones and thus are not ideal to be used for congestion avoidance, their reports can help find discrepancies between the V2V congestion advertisements and congestion information on the cloud, identifying inaccurate and probably malicious congestion measurements. These measurements would then be ignored and the vehicles that originally advertised them would be reported to authorities for eviction.

Finally, if a vehicle requests congestion information about not only the roads that it did not visit but also a few roads that it directly observed and has ground truth for, then the vehicular bots would have a problem. Unless they can distinguish between the requests that the requester already knows a proper answer for and those that the requester knows nothing about, they would probably lie about both. If the requester observes that it receives inaccurate congestion information from a particular vehicle concerning the roads it already knows about, it would then consider this vehicle likely to be a vehicular bot, ignore all advertisements from the vehicle, and notify authorities for the vehicle's eviction.

5.2 BOTVEILLANCE

While BOTVEILLANCE is designed to be resilient to and have a high tracking accuracy despite the challenging network conditions and privacy configurations, we still suggest further improvements on our attack so that future researchers can evaluate their defenses possibly with a more powerful tracking attack. The improvements can be made on our mobility analysis, context-linking mechanism and construction of the tracked car's route.

Our current mobility analysis exploits the observation that the mobility of a vehicle is limited by the mobility of the vehicles in front of it. It could be improved by incorporating a machine-learning approach, which learns the natural driver behavior and predicts the vehicle's mobility based on that, as a supplement to our heuristic.

Based on the road characteristics, we choose a fixed radius for the estimation area where context-linking attempts take place. Estimation errors do not affect the value of this radius. Yet, the context-linking accuracy could be further improved by using an adaptive estimation area radius, which is adjusted based on estimation errors. For example, the radius could be increased if the tracked car is not found in the area, or decreased if there are too many possible candidates with the same dimensions as the tracked car.

The percentage of route—one of the two accuracy metrics for our attack—means the percentage of the tracked car's route that we were able to construct in a single tracking attack. During another attack on the same tracked car, the vehicular bots are very likely to construct a different subset of the tracked car's route due to the randomness of pseudonym changes and silent periods. Therefore, one could construct the entire route of the tracked car by merging the partial routes obtained from multiple tracking attacks.

The attack surface considered by earlier work on location privacy is limited. The first implementation of the all-seeing global attacker model—BOTVEILLANCE—provides a better and more comprehensive understanding of the vulnerabilities that need to be addressed for improved location privacy. Here we suggest some improvements on the pseudonym changing schemes and VANET standards in light of the findings from our work.

Though vehicle dimension information seems not to be an identifier that must be changed along with the pseudonyms, our attack showed that it can be used as one. Therefore, in order to prevent tracking using this information, it must be changed with other identifiers without impacting its use in VANET safety distance calculations—possibly by each vehicle advertising dimensions that alternate between different sets of values, all of which are greater than its actual dimensions. This way, a vehicle's dimensions could not be used for context-linking, while each calculated safety distance to the vehicle would still be acceptable. Alternatively, the centimeter granularity of the size field in BSMs could be decreased—reducing the variations in sizes of different cars—or each car could advertise a range of size, provided that cars with similar sizes advertise the same range. Determining the acceptable variations that would be adequate for location privacy, while still meeting traffic safety guarantees, is important future work.

Mobility analysis is a common approach for context-linking after the silent periods vehicles enter following their pseudonym changes. However, vehicles alter their mobility patterns during silent periods to confuse attackers. We improve our mobility analysis with the observation that a vehicle's mobility has to be somewhat similar to the mobility of other vehicles in front of it. A significant improvement on the pseudonym changing schemes could be each vehicle determining a random mobility pattern—which is unnaturally different than the current common mobility pattern on the road—before it enters the silent period. This would increase the frequency of vehicular bots context-linking to the wrong vehicle.

Silent periods have a considerable impact on tracking accuracy; however, there is always a tradeoff between the level of privacy and traffic safety. As we discussed before, long silent periods or silence in places like intersections endanger traffic safety. As a result, vehicles would have to wake up from their silent periods. For an effective defense against a wide variety of tracking attacks, one could design a special broadcasting mechanism that can ensure traffic safety without vehicles ending their silent periods prematurely. For example, vehicles could encrypt and send their BSMs directly to an RSU while in silent period. The RSU could then decrypt the BSMs, authenticate their senders and broadcast them on behalf of the vehicles after removing the pseudonyms from them. This way, the anonymized BSMs could not be used for tracking and traffic safety could be ensured without sacrificing the level of privacy that comes with silent periods. Of course, this solution depends on the assumption that RSUs are both present and not compromised.

We stated earlier that using one pseudonym for a group of vehicles rather than a separate one for each individual vehicle makes context-linking in tracking attacks challenging. BOTVEILLANCE can still track a vehicle in case of group pseudonyms, while all the other tracking attacks that are described in Section 3.3.1 fail; however, it will certainly have an adverse effect on the tracking accuracy of our attack. The reason is that group pseudonyms introduce new edge cases. For example, if there is another vehicle with the same dimensions as the tracked car in the same group, during a silent period, they could switch places by the tracked car slowing down to the other car's expected position and the other car speeding up to the tracked car's old expected position or vice versa. Alternatively, there might be a situation where the tracked car is joining another group while some other vehicle with the same dimensions is joining the tracked car's old group at the same location. In these cases, our context-linking algorithm might get confused and start tracking the wrong vehicle. However, as we discussed before, implementing the accountability requirement of pseudonymous systems for group pseudonyms is very challenging, requiring further research.

5.3 RIoT

RIoT is designed in an adaptive manner, and thus is easily extensible with new and possibly more effective exploits. With future security evaluations on IoT devices, our exploit database could be extended with new vulnerabilities in order for possibly a higher effectiveness. Also, for a more comprehensive exploit database, the security of a wider variety of devices under each IoT device category could be assessed. We showed that RIoT can have devastating consequences if not defended against. Thus, in addition to these improvements on our attack, we also suggest some possible countermeasures against it.

Vehicular bots compromise IoT devices by delivering exploit payloads to them directly over a wireless ad hoc connection. Therefore, a honeynet approach could be used for detecting the vehicular bots. When a vehicular bot attempts to compromise an IoT device that is a honeynet node, it could notify all the other devices over the Internet so that they ignore any message from this vehicle.

Since IoT devices that communicate among themselves over a wireless ad hoc connection are generally very close to each other—mostly even in the same building—they could use a localization mechanisms, like INTERLOC, to cooperatively pinpoint locations of senders, and ignore everything sent by nodes more than a threshold away.

If IoT devices have the sufficient computing resources, they could exchange a session key using a direct connection over the Internet with each other, and encrypt everything sent over their wireless ad hoc connections with this key. Then, anything that is not encrypted with this key—which would be challenging for vehicular bots to intercept—would be ignored, preventing possible deliveries of exploit payloads.

5.4 SHIELDNET

We already suggested some improvements on the VANET standards, and several techniques to improve the accuracy of the anomaly detection in GPS errors as future work in Section 4.5. These recommendations need to be implemented in order for any detection mechanism to successfully identify vehicular bots using their botnet communication protocol. In addition, we also recommended a mechanism that dynamically determines the best threshold values for SHIELDNET's *temporal sequencer* based on changing network activity in order to achieve a better vehicular bot identification accuracy. In this section, we discuss a defense mechanism that could supplement SHIELDNET.

Some VANET-analog of honeynets—another popular approach for detecting and analyzing botnets [McC03] [The05] [YBP05]—might be helpful in detecting vehicular botnets, perhaps especially if the system includes non-vehicular components that can observe large quantities of the VANET traffic. If vehicular botnets spread primarily by infected cars coming physically close to uninfected cars, at which point a compromise attempt is made, RSUs could pose as vehicles for honeynet purposes. If they are installed in enough locations, the infected cars are likely to drive past one of them, which could then capture the vehicular bot. Given that an RSU has a much larger communication range than a vehicle, achieving a vehicular honeynet with real vehicles as effective as the one with RSUs would require a much larger number of vehicles and thus be cost prohibitive. On the other hand, if the RSU can be distinguished from a genuine vehicle by the vehicular botnet software, it might avoid attempting to infect this honeynet "vehicle". Internet botnet designers face an analogous challenge, since they would prefer not to have their bots captured in a honeynet. Thus, they take measures to learn the locations of honeynets and to differentiate between honeynet nodes and ordinary nodes. In turn, the honeynet designers take measures to deceive the bots. One can imagine a similar competition between vehicular botnet designers and vehicular honeynet designers.

5.5 Possible Future Vehicular Botnet Attacks

In order to demonstrate the dangers and effectiveness of our vehicular botnets, we presented three powerful vehicular botnet attacks in this dissertation. However, there are many other possible vehicular botnet attacks that the future may bring. Here we discuss some of these possibilities, which can be categorized under disruptive, criminal and economic incentives.

A vehicular botnet attack aiming to cause accidents in platooning situations could be an example of a disruptive incentive. A platoon is a group of autonomous vehicles that travel at constant speed with a very short distance from each other by using Cooperative Adaptive Cruise Control (CACC) [SJB14]. Platooning can provide a better traffic safety, increase travel speeds, save fuel, and decrease pollution. However, CACC mechanisms are highly dependent on the availability and correctness of the information being exchanged among platooning vehicles via inter-vehicular communication; [SBJ14] shows that the system becomes unstable if the inputs to CACC are not corrected within 300 milliseconds. Therefore, vehicular bots could cause accidents in platooning situations if they disrupt the information propagation within a platoon by jamming it at several locations. Alternatively, they could create one or more ghost vehicles that join a platoon by manufacturing all the BSMs and other messages that may be required by the CACC mechanism in order to disrupt it from within; they could also join a platoon themselves to do it. Any such disruption to CACC—through either jamming from outside, or not relaying messages or relaying malicious ones within a platoon—would impact the coordination needed among vehicles to travel within a very short distance of each other, and might cause a chain reaction accident.

Some vehicular botnet attacks could be carried out for criminal incentives. For example, vehicular bots could help criminals to escape police pursuit by making subtle timely lane changes and driving slowly in front of police cars in order to temporarily block them. Similarly, they could also delay the arrival of police at a crime scene to provide criminals with a chance to escape, or delay emergency vehicles to maximize the effect of a crime such as arson. Another example could be using vehicular bots to aid in accident insurance fraud. In such fraud, someone claims to have been hit by a car and files an insurance claim. Often, they are intentionally lightly brushed by a car to give credence to the claim. Vehicular bots could act to cause precisely this kind of accident, damaging their owners, or could force other vehicles to do so.

Finally, attackers might choose to use vehicular bots for economic incentives. For example, vehicular bots could favor one ride sharing service over others by clearing paths for the preferred service and blocking them for its competitors. This could be achieved by using a modified version of our congestion attack. Similarly, using another modified version of our attack, vehicular bots could manipulate the routes taken by vehicles to make them pass by favored stores.

Possible future vehicular botnet attacks are not limited to the ones discussed in this section. The collective power of vehicular bots will enable many more attacks that are not possible for a single malicious vehicle to perform. Therefore, the attack surface of VANETs widened by this dissertation needs to be reevaluated as future work so that we can be prepared for the dangers that vehicular botnets could bring to reality.

REFERENCES

[ABI16] ABI Research. "Enterprise Wearable Scanner and Reader Technologies: Devices, Use Cases, and Supplier Ecosystem Analysis." https://www.abiresearch.com/market-research/product/1025688-enterprise-wearable-scanner-and-reader-tec/, 2016.

[Ame19] American Hospital Association. "Fast Facts on U.S. Hospitals." https://www.aha.org/statistics/fast-facts-us-hospitals, 2019.

[BAG12] Igor Bilogrevic, Imad Aad, Philip Ginzboorg, Valtteri Niemi, and Jean Pierre Hubaux. "Track Me If You Can: On the Effectiveness of Context-based Identifier Changes in Deployed Mobile Networks." In *NDSS*, 2012.

[BCJ09] Michael Bailey, Evan Cooke, Farnam Jahanian, Yunjing Xu, and Manish Karir. "A Survey of Botnet Technology and Defenses." In *IEEE CATCH'09*, 2009.

[BGS10] Ramon Bauza, Javier Gozalvez, and Joaquin Sanchez-Soriano. "Road Traffic Congestion Detection Through Cooperative Vehicle-to-Vehicle Communications." In *IEEE 35th Conference on Local Computer Networks*, 2010.

[BHV07] Levente Buttyán, Tamás Holczer, and István Vajda. "On the Effectiveness of Changing Pseudonyms to Provide Location Privacy in VANETs." In *ESAS*, 2007.

[BHW09] Levente Buttyán, Tamás Holczer, André Weimerskirch, and William Whyte. "SLOW: A Practical Pseudonym Changing Scheme for Location Privacy in VANETs." In *IEEE VNC*, 2009.

[BKK11] Leyla Bilge, Engin Kirda, Christopher Kruegel, and Marco Balduzzi. "EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis." In *NDSS*, 2011.

[BMB12] Norbert Bismeyer, Sebastian Mauthofer, Kpatcha M Bayarou, and Frank Kargl. "Assessment of Node Trustworthiness in VANETs Using Data Plausibility Checks with Particle Filters." In *IEEE VNC*, 2012.

[BMC08] Mike Burmester, Emmanouil Magkos, and Vassilis Chrissikopoulos. "Strengthening Privacy Protection in VANETs." In *IEEE WIMOB*, 2008.

[BNP12] Norbert Bismeyer, Joël Njeukam, Jonathan Petit, and Kpatcha M Bayarou. "Central Misbehavior Evaluation for VANETs based on Mobility Data Plausibility." In *ACM VANET*, 2012.

[BS03] Alastair R Beresford and Frank Stajano. "Location Privacy in Pervasive Computing." *IEEE Pervasive Computing*, **2**(1):46–55, 2003.

[BSB10] Norbert Bismeyer, Christian Stresing, and Kpatcha M Bayarou. "Intrusion Detection in VANETs Through Verification of Vehicle Movement Data." In *IEEE VNC*, 2010.

[BW15] Mario Ballano Barcena and Candid Wueest. “Insecurity in the Internet of Things.” Technical report, Security Response, Symantec, 2015.

[Cai17] Timothy Cain. “2016 Year End U.S. Vehicle Sales Rankings – Top 299 Best-Selling Vehicles In America.” https://www.goodcarbadcar.net/2017/01/usa-2016-vehicle-sales-by-model-manufacturer-brand.html, 2017.

[CBK06] Sergey Chapkin, Boto Bako, Frank Kargl, and Elmar Schoch. “Location Tracking Attack in Ad Hoc Networks based on Topology Information.” In *IEEE MASS*, 2006.

[CFE15] Lara Codeca, Raphael Frank, and Thomas Engel. “Luxembourg SUMO Traffic (LuST) Scenario: 24 Hours of Mobility for Vehicular Networking Research.” In *IEEE VNC*, 2015.

[CFG10] Matteo Cesana, Luigi Fratta, Mario Gerla, Eugenio Giordano, and Giovanni Pau. “C-VeT the UCLA Campus Vehicular Testbed: Integration of VANET and Mesh Networks.” In *IEEE European Wireless Conference*, 2010.

[Cho17] Pramit Choudhary. “Introduction to Anomaly Detection.” https://www.datascience.com/blog/python-anomaly-detection, 2017.

[Cim17] Catalin Cimpanu. “About 90% of Smart TVs Vulnerable to Remote Hacking via Rogue TV Signals.” https://www.bleepingcomputer.com/news/security/about-90-percent-of-smart-tvs-vulnerable-to-remote-hacking-via-rogue-tv-signals/, 2017.

[CMK11] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, and Tadayoshi Kohno. “Comprehensive Experimental Analyses of Automotive Attack Surfaces.” In *20th USENIX Security Symposium*, 2011.

[Col18] Louis Columbus. “2018 Roundup of Internet of Things Forecasts and Market Estimates.” https://www.forbes.com/sites/louiscolumbus/2018/12/13/2018-roundup-of-internet-of-things-forecasts-and-market-estimates/, 2018.

[CPH07] Giorgio Calandriello, Panos Papadimitratos, Jean Pierre Hubaux, and Antonio Lioy. “Efficient and Robust Pseudonymous Authentication in VANET.” In *ACM VANET*, 2007.

[CS16] Ke Wan Ching and Manmeet Mahinderjit Singh. “Wearable Technology Devices Security and Privacy Vulnerability Analysis.” *Int. J. Netw. Secur. Appl*, **8**(3):19–30, 2016.

[CSS10] Chia Yuan Cho, Eui Chul Richard Shin, and Dawn Song. “Inference and Analysis of Formal Models of Botnet Command and Control Protocols.” In *17th ACM Conference on Computer and Communications Security*, 2010.

[Cun13] Wayne Cunningham. “Honda Tech Warns Drivers of Pedestrian Presence.” https://www.cnet.com/news/honda-tech-warns-drivers-of-pedestrian-presence/, 2013.

[CVT09a] Brijesh Chaurasia, Shekhar Verma, G Tomar, and Ajith Abraham. “Optimizing Pseudonym Updation in Vehicular Ad-Hoc Networks.” *Transactions on Computational Science IV*, pp. 136–148, 2009.

[CVT09b] Brijesh Kumar Chaurasia, Shekhar Verma, Geetam Singh Tomar, and SM Bhaskar. “Pseudonym Based Mechanism for Sustaining Privacy in VANETs.” In *IEEE CICSYN*, 2009.

[CZL10] Wenping Chen, Sencun Zhu, and Deying Li. “VAN: Vehicle-assisted Shortest-time Path Navigation.” In *IEEE MASS*, 2010.

[DER08] Tim Dasilva, Kevin Eustice, and Peter Reiher. “Johnny Appleseed: Wardriving to Reduce Interference in Chaotic Wireless Deployments.” In *ACM MSWiM*, 2008.

[DFM05] Florian Dotzer, Lars Fischer, and Przemyslaw Magiera. “VARS: A Vehicle Ad-hoc Network Reputation System.” In *IEEE WoWMoM*, 2005.

[DK05] Matt Duckham and Lars Kulik. “Simulation of Obfuscation and Negotiation for Location Privacy.” In *COSIT*, 2005.

[Eic07] Stephan Eichler. “Strategies for Pseudonym Changes in Vehicular Ad Hoc Networks Depending on Node Mobility.” In *IEEE IV’07*, 2007.

[EKS96] Martin Ester, Hans Peter Kriegel, Jorg Sander, and Xiaowei Xu. “A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.” In *KDD*, 1996.

[EMT02] Magda El Zarki, Sharad Mehrotra, Gene Tsudik, and Nalini Venkatasubramanian. “Security Issues in a Future Vehicular Network.” *European Wireless*, **2**, 2002.

[ESG10] David Eckhoff, Christoph Sommer, Tobias Gansen, Reinhard German, and Falko Dressler. “Strong and Affordable Location Privacy in VANETs: Identity Diffusion Using Time-slots and Swapping.” In *IEEE VNC*, 2010.

[Eva03] Leonard Evans. “A New Traffic Safety Vision for the United States.” *American Journal of Public Health*, **93**(9):1384–1386, 2003.

[Evi08] Evita. “E-safety Vehicle Intrusion Protected Applications.” https://www.evita-project.org, 2008.

[FBR16] Yanick Fratantonio, Antonio Bianchi, William Robertson, Engin Kirda, Christopher Kruegel, and Giovanni Vigna. “Triggerscope: Towards Detecting Logic Bombs in Android Applications.” In *IEEE Symposium on Security and Privacy*, 2016.

[FBS10] Stefano Fontanelli, Enrico Bini, and Paolo Santi. “Dynamic Route Planning in Vehicular Networks based on Future Travel Estimation.” In *IEEE VNC*, 2010.

[Fit13] Michael Fitzgerald. “An Internet for Manufacturing.” https://www.technologyreview.com/s/509331/an-internet-for-manufacturing/, 2013.

[Fra17] Lorenzo Franceschi-Bicchierai. "Nasty Bug Left Thousands of Internet of Things Devices Open to Hackers." https://motherboard.vice.com/en_us/article/gybm4b/internet-of-things-camera-axis-bug, 2017.

[FRF07] Julien Freudiger, Maxim Raya, Márk Félegyházi, Panos Papadimitratos, and Jean Pierre Hubaux. "Mix-zones for Location Privacy in Vehicular Networks." In *ACM WiN-ITS*, 2007.

[FSR09] Maryam Feily, Alireza Shahrestani, and Sureswaran Ramadass. "A Survey of Botnet and Botnet Detection." In *IEEE SECURWARE*, 2009.

[GB08] Gilles Guette and Ciarán Bryce. "Using TPMs to Secure Vehicular Ad-hoc Networks (VANETs)." In *IFIP International Workshop on Information Security Theory and Practices*, 2008.

[GG07] Matthias Gerlach and Felix Guttler. "Privacy in VANETs Using Changing Pseudonyms-Ideal and Real." In *IEEE VTC*, 2007.

[GGR15a] Mevlut Turker Garip, Mehmet Emre Gursoy, Peter Reiher, and Mario Gerla. "Congestion Attacks to Autonomous Cars Using Vehicular Botnets." In *NDSS*, 2015.

[GGR15b] Mevlut Turker Garip, Mehmet Emre Gursoy, Peter Reiher, and Mario Gerla. "Scalable Reactive Vehicle-to-Vehicle Congestion Avoidance Mechanism." In *IEEE CCNC*, 2015.

[GGS04] Philippe Golle, Dan Greene, and Jessica Staddon. "Detecting and Correcting Malicious Data in VANETs." In *ACM VANET*, 2004.

[GKR17] Mevlut Turker Garip, Paul Hyungmin Kim, Peter Reiher, and Mario Gerla. "INTERLOC: An Interference-aware RSSI-based Localization and Sybil Attack Detection Mechanism for Vehicular Ad Hoc Networks." In *IEEE CCNC*, 2017.

[GM17] April Glaser and Rani Molla. "The Number of Robots Sold in the U.S. Will Jump Nearly 300 Percent in Nine Years." https://www.recode.net/2017/4/3/15123006/robots-sold-america-growth-300-percent-jobs-automation, 2017.

[GPZ08] Guofei Gu, Roberto Perdisci, Junjie Zhang, and Wenke Lee. "BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-independent Botnet Detection." In *17th USENIX Security Symposium*, 2008.

[GRG16] Mevlut Turker Garip, Peter Reiher, and Mario Gerla. "GHOST: Concealing Vehicular Botnet Communication in the VANET Control Channel." In *IEEE IWCMC*, 2016.

[GRG18] Mevlut Turker Garip, Peter Reiher, and Mario Gerla. "BOTVEILLANCE: A Vehicular Botnet Surveillance Attack against Pseudonymous Systems in VANETs." In *IFIP WMNC*, 2018.

[Gru15] Vera Gruessner. “Market for Wearable Devices Expected to Double by 2018.” https://mhealthintelligence.com/news/market-for-wearable-devices-expected-to-double-by-2018, 2015.

[HCL04] Jean Pierre Hubaux, Srdjan Capkun, and Jun Luo. “The Security and Privacy of Smart Vehicles.” *IEEE Security & Privacy*, **2**(3):49–55, 2004.

[HGX10] Baik Hoh, Marco Gruteser, Hui Xiong, and Ansaf Alrabady. “Achieving Guaranteed Anonymity in GPS Traces via Uncertainty-aware Path Cloaking.” *IEEE Transactions on Mobile Computing*, **9**(8):1089–1107, 2010.

[Hil18] Michael Hill. “IT Pros: IoT Devices Most Vulnerable to Wi-Fi Attacks.” https://www.infosecurity-magazine.com/news/iot-devices-most-vulnerable-wifi/, 2018.

[HSL13] Pei Jin He, Kuo Feng Ssu, and Yu Yuan Lin. “Sharing Trajectories of Autonomous Driving Vehicles to Achieve Time-efficient Path Navigation.” In *IEEE VNC*, 2013.

[Huf14] Huffington Post. “Chris Christie Traffic Scandal.” https://www.huffingtonpost.com/tag/chris-christie-traffic-scandal/, 2014.

[HYM05] Leping Huang, Hiroshi Yamane, Kanta Matsuura, and Kaoru Sezaki. “Towards Modeling Wireless Location Privacy.” In *International Workshop on Privacy Enhancing Technologies*, 2005.

[IMT10] Rob Millerb Ishtiaq Roufa, Hossen Mustafaa, Sangho Ohb Travis Taylora, Wenyuan Xua, Marco Gruteserb, Wade Trappeb, and Ivan Seskarb. “Security and Privacy Vulnerabilities of In-car Wireless Networks: A Tire Pressure Monitoring System Case Study.” In *19th USENIX Security Symposium*, 2010.

[Int19] Intel Corporation. “A Guide to the Internet of Things Infographic.” https://www.intel.com/content/www/us/en/internet-of-things/infographics/guide-to-iot.html, 2019.

[Jam17] Haseeb Jamal. “Sources of Errors in GPS and Their Correction.” https://www.aboutcivil.org/sources-of-errors-in-gps.html, 2017.

[JGO06] Jungwook Jun, Randall Guensler, and Jennifer Ogle. “Smoothing Methods to Minimize Impact of Global Positioning System Random Error on Travel Distance, Speed, and Acceleration Profile Estimates.” *Transportation Research Record: Journal of the Transportation Research Board*, (1972):141–150, 2006.

[JS17] Anthony James and Moshe Ben Simon. “Medical Device Hijack Cyber Attacks Evolve.” https://www.rsaconference.com/writable/presentations/file_upload/hta-r02-medjack.3-new-research-on-attacks-on-hospital-medical-devices.pdf, 2017.

[Kaf17] Sushant Kafle. “DBSCAN.” https://github.com/SushantKafle/DBSCAN, 2017.

[KEB12] Daniel Krajzewicz, Jakob Erdmann, Michael Behrisch, and Laura Bieker. “Recent Development and Applications of SUMO-simulation of Urban Mobility.” *International Journal on Advances in Systems and Measurements*, **5**(3&4), 2012.

[Ken11] John B Kenney. “Dedicated Short-range Communications (DSRC) Standards in the United States.” *Proceedings of the IEEE*, **99**(7):1162–1182, 2011.

[KG11] Wooseong Kim and Mario Gerla. “NAVOPT: Navigator Assisted Vehicular Route Optimizer.” In *IMIS*, 2011.

[KGS10] Sheila G Klauer, Feng Guo, Jeremy Sudweeks, and Thomas A Dingus. “An Analysis of Driver Inattention Using a Case-crossover Approach on 100-car Data: Final Report.” *U.S. Department of Transportation No. HS-811 334*, 2010.

[KRH07] Anestis Karasaridis, Brian Rexroad, and David Hoeflin. “Wide-scale Botnet Detection and Characterization.” In *First Workshop on Hot Topics in Understanding Botnets*, 2007.

[KSD10] Tiffany Hyun Jin Kim, Ahren Studer, Rituik Dubey, Xin Zhang, Adrian Perrig, Fan Bai, Bhargav Bellur, and Aravind Iyer. “VANET Alert Endorsement Using Multi-source Filters.” In *ACM VANET*, 2010.

[KSM12] Peter Knapik, Elmar Schoch, Maik Müller, and Frank Kargl. “Understanding Vehicle Related Crime to Elaborate on Countermeasures Based on ADAS and V2X Communication.” In *IEEE VNC*, 2012.

[Ley15] John Leyden. “Samsung Smart Fridge Leaves Gmail Logins Open to Attack.” https://www.theregister.co.uk/2015/08/24/smart_fridge_security_fubar/, 2015.

[LL09] Jianxiong Liao and Jianqing Li. “Effectively Changing Pseudonyms for Privacy Protection in VANETs.” In *IEEE ISPAN*, 2009.

[LLG15] Jie Li, Huang Lu, and Mohsen Guizani. “ACPN: A Novel Authentication Framework with Conditional Privacy-preservation and Non-repudiation for VANETs.” *IEEE Transactions on Parallel and Distributed Systems*, **26**(4):938–948, 2015.

[LLL11] Rongxing Lu, Xiaodong Lin, Tom H Luan, Xiaohui Liang, and Xuemin Shen. “Anonymity Analysis on Social Spot Based Pseudonym Changing for Location Privacy in VANETs.” In *IEEE ICC*, 2011.

[LLL12] Rongxing Lu, Xiaodong Lin, Tom H Luan, Xiaohui Liang, and Xuemin Shen. “Pseudonym Changing at Social Spots: An Effective Strategy for Location Privacy in VANETs.” *IEEE Transactions on Vehicular Technology*, **61**(1):86–96, 2012.

[LLS10] Rongxing Lu, Xiaodong Lin, and Xuemin Shen. “SPRING: A Social-based Privacy-preserving Packet Forwarding Protocol for Vehicular Delay Tolerant Networks.” In *IEEE INFOCOM*, 2010.

[LLX15] Chi Lin, Kun Liu, Bo Xu, Jing Deng, Chang Wu Yu, and Guowei Wu. "VCLT: An Accurate Trajectory Tracking Attack based on Crowdsourcing in VANETs." In *International Conference on Algorithms and Architectures for Parallel Processing*, 2015.

[LLZ08] Rongxing Lu, Xiaodong Lin, Haojin Zhu, P-H Ho, and Xuemin Shen. "ECPP: Efficient Conditional Privacy Preservation Protocol for Secure Vehicular Communications." In *IEEE INFOCOM*, 2008.

[LMM11] Ilias Leontiadis, Gustavo Marfia, David Mack, Giovanni Pau, Cecilia Mascolo, and Mario Gerla. "On the Effectiveness of an Opportunistic Traffic Management System for Vehicular Networks." *IEEE Transactions on ITS*, **12**(4):1537–1548, 2011.

[LPA06] Uichin Lee, Joon Sang Park, Eyal Amir, and Mario Gerla. "FleaNet: A Virtual Market Place on Vehicular Networks." In *Third Annual International Conference on Mobile and Ubiquitous Systems: Networking* & *Services*, 2006.

[LPB13] Stéphanie Lefevre, Jonathan Petit, Ruzena Bajcsy, Christian Laugier, and Frank Kargl. "Impact of V2X Privacy Strategies on Intersection Collision Avoidance Systems." In *IEEE VNC*, 2013.

[LSH06] Mingyan Li, Krishna Sampigethaya, Leping Huang, and Radha Poovendran. "Swing & Swap: User-centric Approaches Towards Maximizing Location Privacy." In *ACM Workshop on Privacy in Electronic Society*, 2006.

[LSK06] Tim Leinmuller, Elmar Schoch, and Frank Kargl. "Position Verification Approaches for Vehicular Ad Hoc Networks." *IEEE Wireless Communications*, **13**(5):16–21, 2006.

[Mar17] Chuck Martin. "Smart Door Lock Sales Heading To $357 Million." https://www.mediapost.com/publications/article/305118/smart-door-lock-sales-heading-to-357-million.html, 2017.

[Mar18] Peter Martini. "Hackers Graduate to Financial Gain as Motivation for IoT Attacks." https://www.itproportal.com/features/hackers-graduate-to-financial-gain-as-motivation-for-iot-attacks/, 2018.

[McC03] Bill McCarty. "Botnets: Big and Bigger." *IEEE Security* & *Privacy*, **1**(4):87–90, 2003.

[Mil14] Blake Miller. "Magnetic Compass Accuracy, Sighting, and Triangulation." https://sectionhiker.com/magnetic-compass-accuracy/, 2014.

[MQP17] Federico Maggi, Davide Quarta, Marcello Pogliani, Mario Polino, Andrea M Zanchettin, and Stefano Zanero. "Rogue Robots: Testing the Limits of an Industrial Robot's Security." Technical report, Technical report, Trend Micro, Politecnico di Milano, 2017.

[MR13] Milos Milojevic and Veselin Rakocevic. "Distributed Vehicular Traffic Congestion Detection Algorithm for Urban Environments." In *IEEE VNC*, 2013.

[MRC08] Tyler Moore, Maxim Raya, Jolyon Clulow, Panos Papadimitratos, Ross Anderson, and Jean Pierre Hubaux. "Fast Exclusion of Errant Devices from Vehicular Networks." In *IEEE SECON*, 2008.

[MV13] Raquel Machado and Krishna Venkatasubramanian. "Establishing Trust in a Vehicular Network." In *IEEE VNC*, 2013.

[NAP13] Yacin Nadji, Manos Antonakakis, Roberto Perdisci, David Dagon, and Wenke Lee. "Beheading Hydras: Performing Effective Botnet Takedowns." In *ACM SIGSAC Conference on Computer & Communications Security*, 2013.

[Nat15] National Association of Manufacturers. "United States Manufacturing Facts." https://www.nam.org/Data-and-Reports/State-Manufacturing-Data/2014-State-Manufacturing-Data/Manufacturing-Facts–United-States/, 2015.

[Nav12] Navigant Research. "Executive Summary Smart Appliances." https://www.navigantresearch.com/wp-content/uploads/2012/09/SAPP-12-Executive-Summary.pdf, 2012.

[New17] Lily Hay Newman. "Medical Devices Are The Next Security Nightmare." https://www.wired.com/2017/03/medical-devices-next-security-nightmare/, 2017.

[Ngu17] Nicole Nguyen. "If You Have a Smart TV, Take a Closer Look at Your Privacy Settings." https://www.cnbc.com/2017/03/09/if-you-have-a-smart-tv-take-a-closer-look-at-your-privacy-settings.html, 2017.

[OGZ97] Wilson Odero, P Garner, and A Zwi. "Road Traffic Injuries in Developing Countries: a Comprehensive Review of Epidemiological Studies." *Tropical Medicine & International Health*, **2**(5):445–460, 1997.

[Ope15] OpenSim Ltd. "OMNeT++ Discrete Event Simulator." https://www.omnetpp.org, 2015.

[Pal17] Danny Palmer. "175,000 IoT Cameras Can Be Remotely Hacked Thanks to Flaw, Says Security Researcher." https://www.zdnet.com/article/175000-iot-cameras-can-be-remotely-hacked-thanks-to-flaw-says-security-researcher/, 2017.

[PBF15] Jonathan Petit, Djurrre Broekhuis, Michael Feiri, and Frank Kargl. "Connected Vehicles: Surveillance Threat and Mitigation." In *Black Hat Europe*, 2015.

[Per18] Sarah Perez. "47.3 Million U.S. Adults Have Access to a Smart Speaker, Report Says." https://techcrunch.com/2018/03/07/47-3-million-u-s-adults-have-access-to-a-smart-speaker-report-says/, 2018.

[PFK11] Jonathan Petit, Michael Feiri, and Frank Kargl. "Spoofed Data Detection in VANETs Using Dynamic Thresholds." In *IEEE VNC*, 2011.

[PFK14] Jonathan Petit, Michael Feiri, and Frank Kargl. "Revisiting Attacker Model for Smart Vehicles." In *IEEE WiVeC*, 2014.

[Pio17] Ekaterina Pioryshkina. "What Every Retail CMO Should Know About Beacons and Proximity Marketing." https://www.iflexion.com/blog/every-retail-cmo-know-beacons-proximity-marketing/, 2017.

[Pla19] Plant Automation Technology. "Types of Sensors Used in Industrial Automation." https://www.plantautomation-technology.com/articles/types-of-sensors-used-in-industrial-automation, 2019.

[PLF11] Yuanyuan Pan, Jianqing Li, Li Feng, and Ben Xu. "An Analytical Model for Random Changing Pseudonyms Scheme in VANETs." In *IEEE NCIS*, 2011.

[PP05] Bryan Parno and Adrian Perrig. "Challenges in Securing Vehicular Networks." In *ACM Hotnets IV*, 2005.

[PSF15] Jonathan Petit, Florian Schaub, Michael Feiri, and Frank Kargl. "Pseudonym Schemes in Vehicular Networks: A Survey." *IEEE Communications Surveys & Tutorials*, **17**(1):228–255, 2015.

[RBZ11] David Antolino Rivas, José M Barceló-Ordinas, Manel Guerrero Zapata, and Julián D Morillo-Pozo. "Security on VANETs: Privacy, Misbehaving Nodes, False Information and Secure Data Aggregation." *Journal of Network and Computer Applications*, **34**(6):1942–1955, 2011.

[RH07] Maxim Raya and Jean Pierre Hubaux. "Securing Vehicular Ad Hoc Networks." *Journal of Computer Security*, **15**(1):39–68, 2007.

[Ros13] Seth Rosenblatt. "Car Hacking Code Released at Defcon." https://www.cnet.com/news/car-hacking-code-released-at-defcon/, 2013.

[RPA07] Maxim Raya, Panagiotis Papadimitratos, Imad Aad, Daniel Jungels, and Jean Pierre Hubaux. "Eviction of Misbehaving and Faulty Nodes in Vehicular Networks." *IEEE JSAC*, **25**(8):1557–1568, 2007.

[SAE09] SAE Standards. "Dedicated Short Range Communications (DSRC) Message Set Dictionary." *WIP Standard J2735*, November 2009.

[SBJ14] Michele Segata, Bastian Bloessl, Stefan Joerer, Christoph Sommer, Mario Gerla, Renato Lo Cigno, and Falko Dressler. "Towards Inter-vehicle Communication Strategies for Platooning Support." In *IEEE Nets4Cars-Fall*, 2014.

[SCC09] Brett Stone-Gross, Marco Cova, Lorenzo Cavallaro, Bob Gilbert, Martin Szydlowski, Richard Kemmerer, Christopher Kruegel, and Giovanni Vigna. "Your Botnet Is My Botnet: Analysis of a Botnet Takeover." In *16th ACM Conference on Computer and Communications Security*, 2009.

[Sch08] David Schrank. *Urban Mobility Report (2004)*. DIANE Publishing, 2008.

[SFH11] Hagen Stübing, Jonas Firl, and Sorin A Huss. "A Two-stage Verification Process for Car-to-X Mobility Data Based on Path Prediction and Probabilistic Maneuver Recognition." In *IEEE VNC*, 2011.

[SGA11] Huwida Said, Mario Guimaraes, Noora Al Mutawa, and Ibtesam Al Awadhi. "Forensics and War-driving on Unsecured Wireless Network." In *IEEE International Conference for Internet Technology and Secured Transactions*, 2011.

[SHL05] Krishna Sampigethaya, Leping Huang, Mingyan Li, Radha Poovendran, Kanta Matsuura, and Kaoru Sezaki. "CARAVAN: Providing Location Privacy for VANET." Technical report, Washington Univ Seattle Dept Of Electrical Engineering, 2005.

[SJB14] Michele Segata, Stefan Joerer, Bastian Bloessl, Christoph Sommer, Falko Dressler, and Renate Lo Cigno. "Plexe: A Platooning Extension for Veins." In *IEEE VNC*, 2014.

[SLH07] Krishna Sampigethaya, Mingyan Li, Leping Huang, and Radha Poovendran. "AMOEBA: Robust Location Privacy Scheme for VANET." *IEEE JSAC*, **25**(8), 2007.

[SLL10] Yipin Sun, Rongxing Lu, Xiaodong Lin, Xuemin Shen, and Jinshu Su. "An Efficient Pseudonymous Authentication Scheme with Strong Privacy Preservation for Vehicular Communications." *IEEE Transactions on Vehicular Technology*, **59**(7):3589–3603, 2010.

[SLS08] Robert K Schmidt, Tim Leinmüller, Elmar Schoch, Albert Held, and Günter Schäfer. "Vehicle Behavior Analysis to Enhance Security in VANETs." In *IEEE V2VCOM*, 2008.

[Smi13] Gerry Smith. "Driverless Car Could Be Hacked By 14-year-old From Indonesia, Senator Warns." https://www.huffingtonpost.com/2013/05/17/driverless-car-hack_n_3292748.html, 2013.

[Som15] Christoph Sommer. "Veins: Vehicles in Network Simulation." https://veins.car2x.org, 2015.

[SSB09] Ahren Studer, Elaine Shi, Fan Bai, and Adrian Perrig. "TACKing Together Efficient Authentication, Revocation, and Privacy in VANETs." In *IEEE SECON*, 2009.

[Sta16] Statista. "Internet of Things (IoT) Connected Devices Installed Base Worldwide from 2015 to 2025 (in Billions)." https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/, 2016.

[SYB15] Richard Dean Strickland, Meng Yuan, Sue Bai, David Webber, and Radovan Miucic. "Vehicle to Pedestrian Communication System and Method." *U.S. Patent Application 14/450,097*, 2015.

[Tan14] Chuck Tannert. "Self-Driving Cars: Inside The Road Revolution." https://www.fastcompany.com/3022489/self-driving-cars-let-go-of-the-wheel, 2014.

[The05] The Honeynet Project and Research Alliance. "Know Your Enemy, Tracking Botnets." https://honeynet.org/papers/bots, 2005.

[Tie18] Andrew Tierney. "Z-shave. Exploiting Z-wave Downgrade Attacks." https://www.pentestpartners.com/security-blog/z-shave-exploiting-z-wave-downgrade-attacks/, 2018.

[Tut14] Andrei Tutu. "Tesla Model S Vulnerable to Cyber Attacks." https://www.autoevolution.com/news/tesla-model-s-vulnerable-to-cyber-attacks-79407.html, 2014.

[UN 16] UN Economic Commission for Europe. "UN Vehicle Regulations." https://www.unece.org/trans/main/wp29/wp29regs21-40.html, 2016.

[Und14] David Undercoffler. "54 Million Self-driving Cars Will Be on the Road by 2035, Study Finds." https://articles.latimes.com/2014/jan/02/autos/la-fi-hy-autos-ihs-autonomous-cars-study-20140102, 2014.

[US08] U.S. Department of Defense. "GPS Standard Positioning Service Performance Standard." https://www.gps.gov/technical/ps/2008-SPS-performance-standard.pdf, 2008.

[US14] U.S. Department of Transportation Federal Highway Administration. "Mitigation Strategies For Design Exceptions." https://safety.fhwa.dot.gov/geometric/pubs/mitigationstrategies/chapter3/3_lanewidth.cfm, 2014.

[Vla09] James Vlahos. "Surveillance Society: New High-Tech Cameras Are Watching You." https://www.popularmechanics.com/military/a2398/4236865/, 2009.

[Wir10] Wireless LAN Working Group. "Wireless Access in Vehicular Environments." *IEEE Standards*, July 2010.

[WMK10] Björn Wiedersheim, Zhendong Ma, Frank Kargl, and Panos Papadimitratos. "Privacy in Inter-vehicular Networks: Why Simple Pseudonym Change is Not Enough." In *IEEE WONS*, 2010.

[WSD14] Marius Wernke, Pavel Skvortsov, Frank Dürr, and Kurt Rothermel. "A Classification of Location Privacy Attacks and Approaches." *Personal and Ubiquitous Computing*, **18**(1):163–175, 2014.

[XYG06] Bin Xiao, Bo Yu, and Chuanshan Gao. "Detection and Localization of Sybil Nodes in VANETs." In *DIWANS'06*, 2006.

[YB09] Yi Yang and Rajive Bagrodia. "Evaluation of VANET-based Advanced Intelligent Transportation Systems." In *ACM VANET*, 2009.

[YBP05] Vinod Yegneswaran, Paul Barford, and Vern Paxson. "Using Honeynets for Internet Situational Awareness." In *ACM Hotnets IV*, 2005.

[YCW07] Gongjun Yan Yan, Gyanesh Choudhary, Michele C Weigle, and Stephan Olariu. "Providing VANET Security Through Active Position Detection." In *ACM VANET*, 2007.

[ZLC08] Zhaosheng Zhu, Guohan Lu, Yan Chen, Zhi Judy Fu, Phil Roberts, and Keesook Han. "Botnet Research Survey." In *IEEE International Computer Software and Applications Conference*, 2008.

[ZZC09] Yang Zhang, Jing Zhao, and Guohong Cao. "Roadcast: A Popularity Aware Content Sharing Scheme in VANETs." *ACM SIGMOBILE Mobile Computing and Communications Review*, **13**(4):1–14, October 2009.