Congestion Attacks to Autonomous Cars Using Vehicular Botnets

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

Abstract—**The increasing popularity and acceptance of VANETs will make the deployment of autonomous vehicles easier and faster since the VANET will reduce dependence on expensive sensors. However, these benefits are counterbalanced by possible security attacks. We demonstrate a VANET-based botnet attack in an autonomous vehicle scenario that can cause serious congestion by targeting hot spot road segments. We show via simulation that the attack can increase the trip times of the cars in the targeted area by orders of magnitude. After 5 minutes, the targeted road becomes completely unusable. More importantly, the effect of such an attack is not confined to a specific hotspot; the congestion can spread to multiple roads and significantly affect the entire urban grid. We show that current countermeasures are not effective, and point to new possible defenses.**

I. Introduction

Traffic safety is a main concern for many countries [24]. Many accidents are caused by insufficient traffic information and by slow driver reaction to local visual and acoustic inputs. VANETs (Vehicular Ad hoc Networks) 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 within radio range. Alternatively, intermediary cars route and forward packets to intended destinations. Communication is peer-to-peer, without 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. 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.

Expanding on this vision, 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 [1]. In this paper, 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 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 [2]. 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 [1]. VANETs will be integrated into these cars to complement their expensive on-board sensing and collision avoidance tools.

In this paper, we will examine one particular benefit of the VANET, namely vehicular congestion avoidance. Congestion is a serious problem for big cities with high populations [29]. Solutions like Google Map 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 traffic 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. However, whenever a system like an autonomous vehicle cedes control of its operations to a networked computer, we introduce new risks of cyberattacks. The network allows attackers some access to the vehicle's systems, and history has shown that any degree of network access carries a risk of complete compromise. The same capabilities that allow an autonomous vehicle to prevent accidents could fall under the control of a malicious attacker. In particular, the V2V radio connectivity of autonomous vehicles will offer attackers opportunities to combine multiple compromised vehicles into botnets of cars, which will lead to other serious security consequences. Further, the ability to remotely steer the bots without a driver and position them in zones of interest is a formidable weapon in the hands of attackers.

The VANET protocols (routing, content discovery, etc.) have been well-studied, but their security requires more investigation. Some attacks involving autonomous vehicles could be fatal, so intense research in VANET security is needed before people begin using them. In recent years, there have been some incidents where hackers succeeded in taking over the system software of autonomous vehicles and were able to control the actions of the cars [3]. Well-known computer security experts Charlie Miller and Christopher Valasek presented their attacks at Defcon 2013 and proved that it is possible to take over total control of autonomous vehicles [4].

One dangerous security issue for autonomous vehicles and

SENT '15, 8 February 2015, San Diego, CA, USA
Copyright 2015 Internet Society, ISBN 1-891562-39-8
http://dx.doi.org/10.14722/sent.2015.23001

VANETs that has not been studied is the possibility of vehicular botnets, i.e., compromised cars that can be orchestrated by a hostile individual or organization to launch damaging attacks to legitimate cars. In this paper, we demonstrate a successful botnet attack to the V2V propagation of traffic information that was just described above. The attack can cause heavy congestion on any chosen road. Since methods for compromising autonomous vehicles have been demonstrated in [3] and [4], we assume that we already have a botnet of compromised vehicles, instead of focusing on how we can create it. Our botnet attack manipulates the common vulnerabilities of existing and/or proposed VANET congestion avoidance mechanisms. While botnets are common in the Internet, botnet attacks in VANETs will be fundamentally different. First, the purely cyber attacks of VANET botnets will produce not only cyber (like in the Internet) but also real physical damage in an urban environment. For example, our botnet attack can subtly increase estimated travel time on carefully selected road segments and by consequence increase trip times globally by orders of magnitude for many vehicles that have been tricked by false congestion advertisements. Directing the attack to secondary road segments can effectively manipulate the routes of many other cars and cause them to congest unexpected intersections.

The main purpose of this paper is not to simply introduce the congestion attack, but rather to use it as an example to demonstrate the power of vehicular botnets. Although vehicular botnets can be used to perform variety of dangerous attacks - some seemingly more attractive than the congestion attack (e.g. stealing the compromised cars themselves) - the reasons for choosing congestion attack as our example are twofold:

- It demonstrates the vehicular botnet's potential to have a *global* impact on traffic conditions, by starting a domino effect of traffic congestion.
- As attackers often perform DDoS attack performed by Internet botnets, only this time road segments are flooded with *physical* traffic. Much like the DDoS attacks in the Internet, the congestion attack can be performed for economic and political incentives. For example, it 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 Section 2, we discuss the related work that has been done on traffic congestion avoidance using VANETs and on VANET security. In Section 3, we describe the common characteristics and vulnerabilities that VANET congestion avoidance mechanisms have. In Section 4, we explain our attack in detail and present simulation examples. In Section 5, we show the success and speed of our attack with experiment results. In Section 6, we recommend some countermeasures against vehicular botnet attacks. In Section 7, we discuss the danger of vehicular botnet and its different applications. In Section 8, we conclude by discussing possible impacts of our attack on VANET design.

II. Related Work

Deployment of VANETs and Vehicle-to-X (V2X) communications can certainly enhance Intelligent Transportation Systems (ITS) and aid traffic congestion in urban environments. [33] shows (through 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. [9] proposes a system based on vehicle-to-vehicle 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 re-routing in response to congestion. [13] 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 [23], 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 the section on congestion avoidance algorithms, since specifics of those algorithms are core to the research described here.

Security aspect of VANETs has received some, but not enough, attention [26]. [25] discusses the challenges in securing VANETs, comparing it to other types of wireless networks. While it usefully lists the limitations of security applications for VANETs, it assumes a majority of honest nodes. Vehicular botnets, however, can either be deployed such that they create a significant majority in a certain vicinity, or located strategically so that they poison enough number of cars that would result in a successful execution of a botnet-originating attack.

[26] proposes using PKI for VANET communications. This solution assures that each message is from a legitimate car and prevents identity spoofing. However, it is not sufficient to defend against attacks in which messages can be cryptographically authenticated (i.e. valid signatures and certificates) but the contents of the messages are forged. 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. [20] discusses potential problems that bogus location data may cause, and possible solutions. [15] detects sybil nodes through position verification using various sensor data. [10], [30] and [32] discuss alternate approaches and filters/sensors to assess plausibility of vehicle movements and position. Centralized approaches can also be effective against similar problems [11]. Since we do not make use of sybil attacks or position faking in our botnet attack (not for our congestion attack, at least), our attack is resistant to these filters; identifying the information that our bots advertise as outliers is difficult. Nevertheless, we paid extra attention to make bots lie under filters' plausibility thresholds since eviction of misbehaving nodes (e.g. by revoking certificates) in a timely manner is possible as described in [27].

More complex and thorough systems can be built by incorporating complementary sensors and filters. Carefully studying the contents of heartbeat (or Context Awareness) messages, plausibility and misbehavior analyses can be conducted. [28] checks not only vehicle movements and sensor/radar verification of claimed position, but also compliance of the messages to network-related parameters (beacon frequency, transmission power and range etc.). [18] takes the responses of other vehicles and surrounding infrastructures (e.g. Road-Side Units) into consideration when making decisions regarding content plausibility. [18], [22] and [28] all propose reputation-based systems to pinpoint the attackers that inject bogus information to the network. These systems defend against the use of blatantly wrong or implausible values, which led our bots to carefully craft their fake congestion information sent.

III. 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 the cars (e.g. 4G/LTE, V2V, etc.). [9], [13] and [23] 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. [16] and [19] 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. [12] and [21] are examples of V2V, totally distributed congestion avoidance approach. These two methods offer different, complementary advantages. [21] 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, [12] 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 the botnet attack will exploit: they completely trust and forward congestion information without trying to detect outliers or incorrect information. We ran the botnet attacks on functionalities of 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 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. In the subsequent sections, we describe the implementation, how cars store congestion information and how they choose the least congested routes.

A. Message Types and Information Exchange

For the communication among all cars, we assume standard signal range of the 802.11p protocol, which is 300 meters [31]. Two types of messages are implemented in our congestion avoidance simulator: congestion request and congestion response messages. These are in addition to the Basic Safety Messages (BSM) that are regularly sent to nearby vehicles, as explained in [17] and standardized in [8].

1) Congestion Request Messages: When a car approaches the end of a road and can choose from two or more routes, it will need congestion measurements from other cars 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 destination. The car will choose the path based on the responses it gets.

2) Congestion Response Messages: A congestion response message is sent only when a congestion request is received and there are related entries in the congestion information database of the receiving car, ensuring that no superfluous information is transmitted. The response includes congestion information about the roads of interest available at the receiver.

B. Congestion Information Database

1) Creating and Storing Congestion Information: In order to store and exchange congestion measurements, vehicles make use of *congestion info structs*. Each struct 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: Average of the speed readings from all the cars on the same road with the car that crates this measurement.

Timestamp: Time of the measurement's creation to ensure the freshness of measurements and prioritize most recent ones.

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

DB is an abbreviation for the local congestion 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 will be inserted to the database. The first is when the car creates its own measurement by sampling its own speed, together with the speeds of other cars on the same road it is passing along. The second is when measurements are obtained through congestion responses received from other vehicles.

t_α is a time threshold that determines when a measurement created by the car itself becomes outdated in its congestion information database. Cars prioritize the measurements which 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 the botnet attacks.

3) Least Congested Route Selection: When a car approaches the end of a road, it calculates what its interesting roads are in order to differentiate a less congested route and sends out a congestion request asking for these roads. 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 set of the *k* candidate routes. Each $R \in RSet$ is a series of edges (roads) which are represented by E. Therefore, the selection of the least congested route from $RSet$ is performed using the following calculation:

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

IV. Botnet Attack

To the best of our knowledge, botnet attacks have not been studied in the VANET research field. While Internet botnet attacks are common and serious, vehicular botnets can cause more severe, even fatal, damage. Botnets can be used to exploit VANETs in many ways; in this paper, however, we focus only on causing heavy congestion on a previously targeted road. We performed our attack on the implementation of functionalities of the congestion avoidance mechanisms referenced earlier. We concentrate on demonstrating the feasibility of a vehicular botnet attack and leave the design of a secure congestion avoidance mechanism for future research.

A. Types of Vehicles in the Experiment

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

1) Parked Bot Cars: The attackers control a certain number of autonomous cars spread all around the map. Moreover, these cars can drive themselves in a pure robotic fashion (without the driver on board), as some visionaries foresee [1] [2]. These bots are parked in random places and perform malicious advertisements of bogus congestion information if they are close to the area being targeted. 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 bots are in that region along with their positions. During the attack, when parked bots sense that certain portions of the map are not sufficiently covered by compromised cars, they will coordinate with each other and will be directed to cover those areas. They also advertise their congestion measurements with "significantly fresh" timestamps. Within due time, their advertisements will dominate the responses from legitimate cars, which are not part of the botnet.

2) Compromised Cars: These cars have been compromised by attackers using the vulnerabilities present in automated vehicles [3] [4], yet they have legitimate drivers on board, who have no idea that their cars have been compromised. They have legitimate routes and minimize their total trip time like other uninfected cars 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 nor any knowledge of what congestion information their cars create and exchange, these cars are used to advertise malicious congestion data in our attack. They still request congestion information from other cars 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.

3) Uninfected Cars: These cars are the victims in our attack; they honestly follow the congestion avoidance protocol in hopes of minimizing their trip times. They share the congestion measurements in their databases, which are either generated by themselves or heard from other cars, without any malicious alteration. Considering that bots' advertisements can overwhelm legitimate congestion information (both in terms of number and freshness of measurements), uninfected cars will "honestly" disseminate the malicious advertisements originating from botnet members. This will speed up the effect of our attack. Examples of this behavior will be given in the next sections, together with experimental results that show higher percentages of botnet members can overwhelm legitimate information faster and more effectively.

B. Congestion Attack Mechanism

The attack starts with a road targeted to cause the congestion; this is given as a parameter to our simulator at configuration time. Parked bot cars are scattered around the map, inactive and parked. There do not need to be a lot of parked bots for our attack to succeed since the attack does not require their existence or continuous activity (Especially for smaller grids/maps, we saw that practically no parked bots are necessary to execute a catastrophic attack). They turn on only when there are not enough compromised cars to cover the targeted area (number of compromised vehicles required is dynamic, and based on the size of the area, wireless ranges of these vehicles and their mobility patterns). Then, they move towards the targeted area, cover the uncovered parts and begin advertising bogus 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 experiments showed that the heavy congestion on the targeted road caused by our attack also spreads over to neighboring roads, making them unusable.

Bots are allowed to create and advertise congestion information for roads that they haven't been on. Therefore, this manipulative high coverage of their responses causes the bogus measurements to disseminate and dominate very quickly. However, they stay under the defensive filters' thresholds to be credible while still being manipulative [10] [11] [18]. Bots also consider their distances to the roads that their bogus measurements belong to. When bots lie about the timestamps, they need to be old enough to be plausible while still being fresh enough to be prioritized. Furthermore, bots can respond to different cars with different congestion information based on their locations in order to better manipulate the victim cars into choosing a route that passes over the targeted road.

1) Exploiting Basic Safety Messages: Cars broadcast their current speeds at a constant frequency as a part of the mandatory "Basic Safety Messages" (BSM) standardized in [8]. These speed exchanges are used by each car in the congestion avoidance mechanism to calculate the total average speeds on the roads that they travel on. These average speed measurements are then used for travel time calculations and least-congested route selection. 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 using the targeted road will start rerouting themselves, which will prevent the congestion from becoming heavier.

To overcome this effect, bots will advertise high speeds while they are 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.

2) Exploiting Congestion Response Messages: Malicious advertisers generate congestion response messages based on the content of the congestion requests. Manipulation strategies of the bots will be different depending on the requester. There are two different scenarios: requester has the targeted road in its candidate routes, or it is not considering the targeted road.

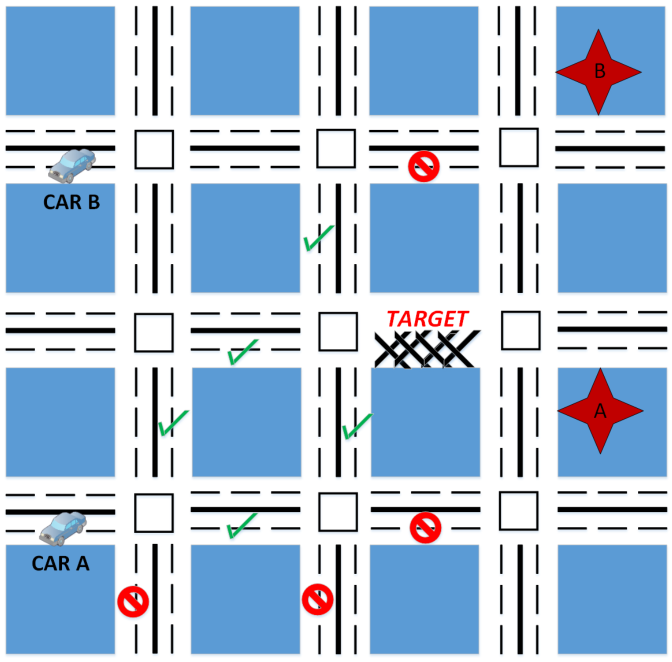

Fig. 1. Malicious advertisements for the cars that are considering taking the targeted road

In Figure 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 victim car A into the target. The congestion responses from the bot cars 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 speeds for such roads, 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 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 bots advertise the roads as congested or not congested is demonstrated in Figure 1.

Another interesting characteristic of the attack can be explained by including the victim car B in the scenario in Figure 1. Consider that right after victim car A chooses its route based on the congestion information it has received from the other cars, victim car B approaches the targeted road from the top left side of the map. Even if all the bots are out of the communication range of B when it sends a congestion request message, A will be able to respond to B with all the bogus congestion information it has just received from the bots. Hence, the malicious advertisements sent by the bots in order to trick A will also exploit the route selection of B, even if there is no bot within its range. We designed and specifically optimized the algorithm that creates bogus information so that previously disseminated information can influence other victims as well.

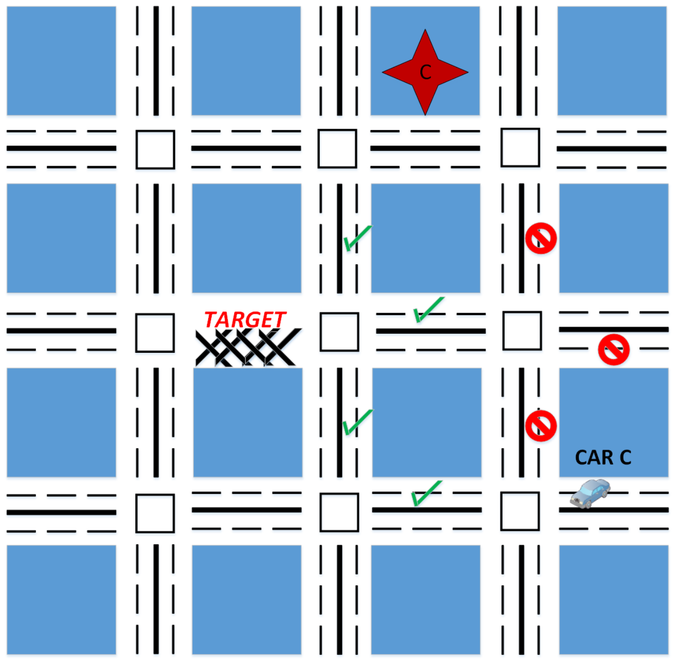

Fig. 2. Malicious advertisements for the cars that are not considering taking the targeted road

Assume that victim car "C" in Figure 2 is not considering taking the targeted road for any of its candidate routes. The bots will then push C towards end of the targeted road as much as possible by advertising high average speeds for the roads that take it closer to the end point of the targeted road, and lower average speeds for the ones that get it further away. If we push as many cars as possible towards the end point, the cars that try to exit the targeted road must 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 car will not create a congestion measurement for a street before exiting that street. Since cars are sitting in the congested areas, they cannot finish traveling on streets as often. Hence, less number of legitimate measurements are created, while bots can further increase their domination in terms of the quantity (and possibly, freshness) of their fake measurements.

V. Evaluation

We used Veins [5] (which combines the SUMO and OMNeT simulators) to conduct VANET experiments. SUMO is an open source traffic simulator for large road networks [6]. We used SUMO to simulate vehicular traffic with random start and end points. This randomization is crucial in order to simulate a realistic traffic and ensure that our attack is not only effective for a specific scenario. Since SUMO does not simulate communication events between cars, we used the OMNeT network simulator [7] with 802.11p integration [31].

In the experiment results, graphs show that metrics are sampled over a timeframe of 2400 seconds. It is important to note here that this time interval is in units of SUMO simulation seconds, which corresponds to a much larger time interval in real life (an experiment that simulates 2400 SUMO seconds 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 bots over the total number of vehicles (1%, 5%, 10% and 20%). We randomize each car's initial and destination points 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 number of cars on the targeted road over time, and 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 overall trip times as a metric to show that our attack has significant effect on both local and global traffic. We implemented our 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, 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 values as low as 0 mph, meaning the 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.

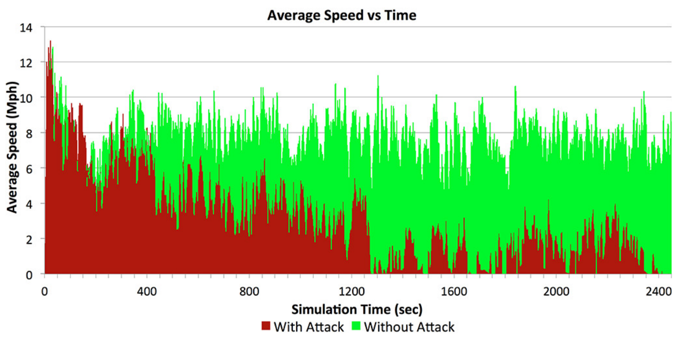

Fig. 3. Average speed on the targeted area with and without active attack

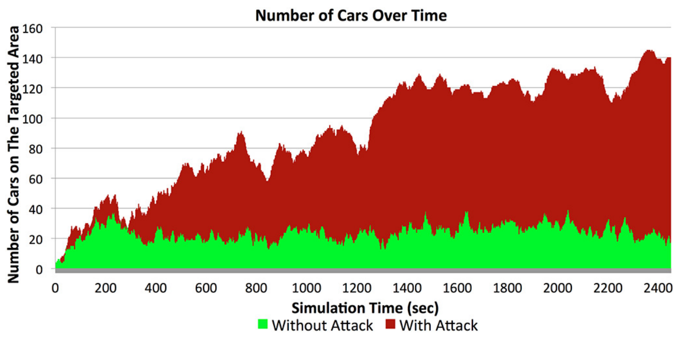

Fig. 4. Number of cars on the targeted area with and without active attack

Figure 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 cars on the targeted area increased by orders of magnitude for the latter case. Figure 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 by which trapped cars can leave. As a result, we observe a consistent increase in the number of cars, as shown in Figure 4, resulting in a highly congested road.

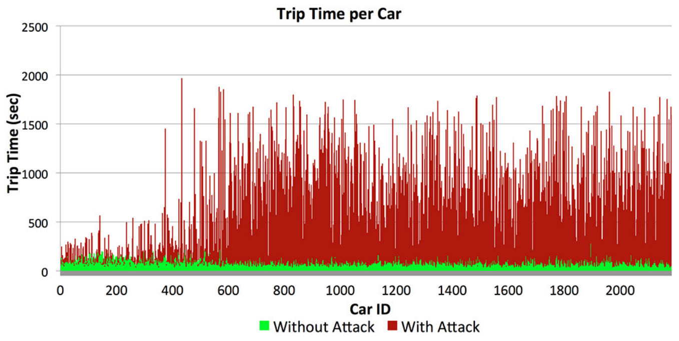

Fig. 5. Trip times for all the cars on the map with and without active attack

In another experiment, we showed that our attack impacts every car on the entire map, not just those around the targeted road. Figure 5 demonstrates the success of our attack in increasing the overall trip times of all the cars in the simulation. This experiment reveals that the local congestion our attack causes will also affect the cars not using the targeted road. For most of the cars, 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 in which a car reaches its destination in 10 hours instead of 1 hour, which indicates a very severe congestion level.

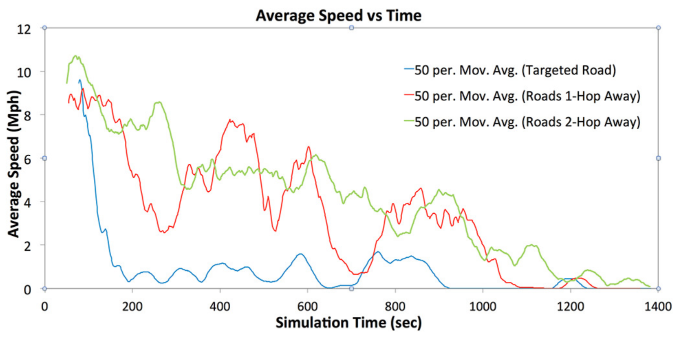

Fig. 6. Average speed on the targeted road and neighboring roads

Figure 6 demonstrates how fast the congestion that is caused by the attack expands to the surrounding roads. The graph shows the average speeds on the targeted road, and the average of all the average speeds of the roads that are 1 or 2 hops away from the targeted road. They follow the same decrease pattern in average speeds with the targeted road. With around 200 simulation seconds delay between them, they all become completely unusable. While we only include 2 hops around the targeted road, we observed that the congestion also expands to the further roads than 2 hops in the same fashion.

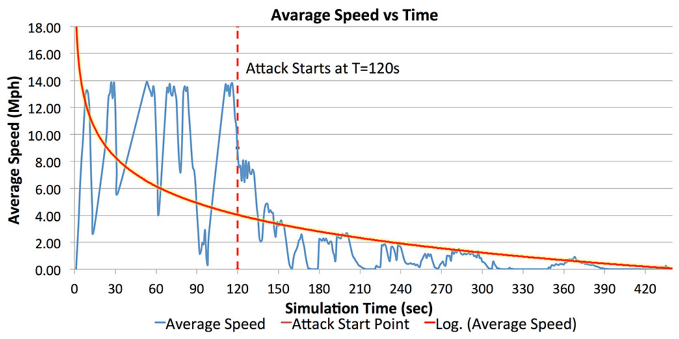

Fig. 7. Average speed graph where attack starts 120 seconds after the beginning of the experiment

The attack shows its effect very quickly. In Figure 7, the attack starts at the 120th simulation second, and in a very few seconds, we observed the severe drop in the average speed on the targeted road. The graph followed a continuous decrease on the logarithmic trend line, drawn in Figure 7, until it reaches zero values. The speed of the attack can vary depending on how far from the beginning of the simulation we start our attack; the later the attack 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.

Let β be the probability of a car to be a compromised bot car and N be the total number of cars generated by SUMO; there will be $\beta \times N$ compromised bot cars and $(1-\beta) \times N$ uninfected cars created throughout the simulation. Figure 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 area in our botnet can increase the average trip time by 50%. With higher β values, the damage caused by our attack reaches even more severe levels (increase on the average trip times grows exponentially when β values are increased linearly).

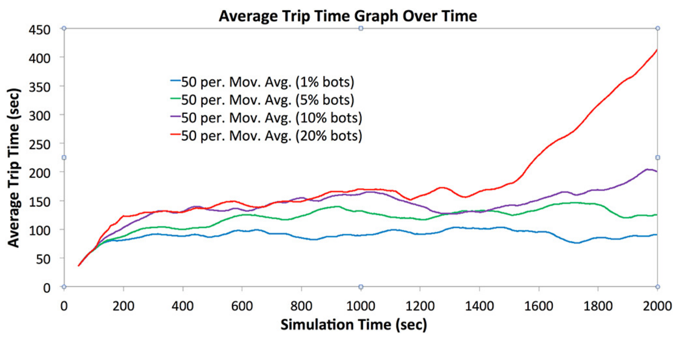

Fig. 8. The graph that shows average of all the cars' trip times over time

VI. Possible Countermeasures

The typical protection from bogus information in VANETs is correlation of messages from neighbors [18]. The bogus messages are often generated by a minority of neighbors, thus can be easily isolated and purged. The offenders are blacklisted and possibly prosecuted. However, in a vehicular botnet attack, the attacker can maneuver enough bot cars to the target zone to achieve a majority, thus neutralizing the correlation defense.

A future research direction will thus be to design defense mechanisms to protect VANETs from botnet attacks. A reputation-based mechanism may be useful in detecting and isolating bot cars since each advertisement can be traced back to its creator. Bots may advertise congestion levels that are inconsistent with each other in the big picture; therefore, an anomaly detection mechanism can analyze advertisements for inconsistencies. Additionally, bot cars will need to communicate with each other frequently. This higher message frequency could also serve as a sign of bad intent. In this case, network flow analysis might reveal the malicious nodes in VANETs. However, clever bots may reduce their message frequency, at the cost of making the attack slower and possibly less effective. Other unusual aspects of the bots' congestion messages, such as a suspicious freshness or an unlikely ability to observe traffic in varying locations, might also serve as clues of the attack.

The countermeasures mentioned above address only the problem of identifying the bot cars after the attack. It is also important to detect the onset of such an attack before it does the damage. In the attack directed to the V2V congestion avoidance scheme, a possible remedy involves each vehicle simultaneously monitoring both V2V and cloud-based congestion avoidance instructions. Recall that the cloud application (say Google Traffic) receives messages from the vehicles via LTE. These messages are tagged by time and position (GPS or LTE tower triangulation). The bot cars are not asked for speed measurements on a particular road. Thus, they cannot launch the request/response attack so the cloud congestion reports can be viewed as (delayed) ground truth. While the cloud traffic congestion application does not react as quickly to traffic jams as V2V does, a sudden discrepancy between the local traffic database and information from cloud can alert the car that something is abnormal. In that case, local congestion responses could be rejected and the driver could be alerted.

Finally, we must detect and "disinfect" the compromised cars to completely mitigate the problem of botnet attacks. This is a complex problem for Internet bot infections, probably as hard for vehicular botnets too. Automated disinfection of Internet machines without consent of the machine's true owner is often regarded as unsafe, and may be illegal. But existing methods of persuading computer owners to disinfect their own machines have had limited success. However, it is possible that special characteristics of autonomous vehicles will ease the problem. For instance, all such vehicles are likely to be licensed by some local government agency. A provable "untreated" bot infection could be regarded as sufficient grounds to revoke the car's license, giving the owner of the vehicle a strong incentive to disinfect his/her car. A key question here is what constitutes sufficient proof to invoke this sanction; thus, research is needed on what observable signals can be regarded as strong evidence that a car has been infected. Whatever signals are chosen, attackers will attempt to conceal those signals while still compromising and misusing autonomous vehicles, so this sort of research is likely to be ongoing and changing in response to alterations in attacker behavior.

VII. Discussion

In this paper, we have chosen to attack the V2V reactive congestion avoidance request/reply scheme. This scheme is simple and efficient (in terms of avoiding congestion). It is also particularly vulnerable because the car announces its routes and the attacker can pick its victims based on the proposed routes. There are other schemes like [21] in which the car does not request explicit information from other cars nor does it announce its routes. In [21] the car waits for the congestion information to propagate to it via dissemination. The attacker, in this case, cannot target a specific car. However, it can still manipulate the congestion information propagated to the legitimate cars. In particular, it can create "tunnels" (by "blocking" some of the streets) and lead unaware vehicles to congestion traps. As pointed out earlier, cloud-based congestion avoidance schemes will be protected from this kind of attack. However, they are much slower in reacting to and reporting congestion. They also require expensive access to the LTE channel.

The congestion attack is not the only botnet attack to autonomous vehicles in VANETs. For example, autonomous vehicles in the future will travel at very high speeds, with minimal distance from each other like a platoon, to increase highway capacity and reduce air resistance. Front and rear lasers and cameras currently used for advanced cruise control will not be sufficient to maintain stable "platooning" at very high speeds. As a minimum, communications with cars 100 meter ahead will be required [14]. LTE will not be adequate for such communications because of excessive delays; V2V will be required. Unfortunately, this will enable attackers to inject misleading information on the V2V channel by positioning enough bots on the highway, causing catastrophic accidents.

In this paper, we have selected the "congestion avoidance" attack because it is relatively simple, yet can demonstrate the power of the vehicular botnet. For the attack on autonomous platoons, it would be much more difficult to covertly inject "robotic" bots (without driver or passengers) on the highway and drive them at high speed to the same section of the highway without getting suspicious. However, depending on the goal of this attack and specifics of how it would be performed, a few non-autonomous bots in a platoon might be enough to cause a catastrophe. Thus, vehicular botnet attacks on platooning systems are an important area of future research.

VIII. Conclusion

In this paper, we review VANET assisted congestion avoidance mechanisms as a possible target of botnet attack. Autonomous vehicles will be particularly at risk as they will completely rely on computer generated routing instructions. We focus on V2V congestion avoidance schemes and put together an attack scenario that is easy to implement in our simulation. To assess the efficiency of the attack, we evaluate traffic flows and delays in presence or absence of attacks. We believe this is the first botnet attack performed in a VANET. While the attack was performed on a simplified congestion avoidance mechanism, it can be shown that this type of attack applies to virtually all V2V schemes, since in a V2V scheme cars draw the operational information from peers' advertisements, which can be manipulated by the botnet nodes. As autonomous vehicles will play an increasing role in our daily lives in the near future, there is clearly a need for fundamental research on autonomous vehicle security. Possible countermeasures were outlined in the paper. While the botnet attack presented here merely targeted road congestion, future work will address other vehicular botnet attacks that can potentially be much more damaging, even lethal to drivers.

References

[1] http://www.fastcompany.com/3022489/innovation-agents/self-driving-cars-let-go-of-the-wheel/.

[2] http://articles.latimes.com/2014/jan/02/autos/la-fi-hy-autos-ihs-autonomous-cars-study-20140102.

[3] http://www.huffingtonpost.com/2013/05/17/driverless-car-hack_n_3292748.html.

[4] http://news.cnet.com/8301-1009_3-57596847-83/car-hacking-code-released-at-defcon/.

[5] http://veins.car2x.org.

[6] http://sumo-sim.org.

[7] http://www.omnetpp.org.

[8] Dedicated short range communications (dsrc) message set dictionary. *WIP Standard J2735*, November 2009.

[9] R. Bauza, J. Gozalvez, and J. Sanchez-Soriano. Road traffic congestion detection through cooperative vehicle-to-vehicle communications. In *IEEE 35th Conference on Local Computer Networks*, 2010.

[10] N. Bismeyer, S. Mauthofer, K. M. Bayarou, and F. Kargl. Assessment of node trustworthiness in vanets using data plausibility checks with particle filters. In *IEEE VNC*, 2012.

[11] N. Bismeyer, J. Njeukam, J. Petit, and K. M. Bayarou. Central misbehavior evaluation for vanets based on mobility data plausibility. In *ACM VANET*, 2012.

[12] W. Chen, S. Zhu, and D. Li. Van: Vehicle-assisted shortest-time path navigation. In *IEEE MASS*, 2010.

[13] S. Fontanelli, E. Bini, and P. Santi. Dynamic route planning in vehicular networks based on future travel estimation. In *IEEE VNC*, 2010.

[14] M. Forster, et al. A cooperative advanced driver assistance system to mitigate vehicular traffic shock waves. In *IEEE INFOCOM*, 2014.

[15] P. Golle, D. Greene, and J. Staddon. Detecting and correcting malicious data in vanets. In *ACM VANET*, 2004.

[16] P. J. He, K. F. Ssu, and Y. Y. Lin. Sharing trajectories of autonomous driving vehicles to achieve time-efficient path navigation. In *IEEE VNC*, 2013.

[17] J. Kenney. Dedicated short-range communications (dsrc) standards in the united states. *Proceedings of the IEEE*, 99(7):1162–1182, July 2011.

[18] T. H. J. Kim, A. Studer, R. Dubey, and X. Zhang et al. Vanet alert endorsement using multi-source filters. In *ACM VANET*, 2010.

[19] W. Kim and M. Gerla. Navopt: Navigator assisted vehicular route optimizer. In *5th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, 2011.

[20] T. Leinmuller, E. Schoch, and F. Kargl. Position verification approaches for vehicular ad hoc networks. *IEEE Wireless Communications*, 13(5):16–21, 2006.

[21] I. Leontiadis, G. Marfia, D. Mack, G. Pau, C. Mascolo, and M. Gerla. On the effectiveness of an opportunistic traffic management system for vehicular networks. *IEEE Transactions on ITS*, 12(4):1537–1548, 2011.

[22] R. Machado and K. Venkatasubramanian. Establishing trust in a vehicular network. In *IEEE VNC*, 2013.

[23] M. Milojevic and V. Rakocevic. Distributed vehicular traffic congestion detection algorithm for urban environments. In *IEEE VNC*, 2013.

[24] W. 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, May 1997.

[25] B. Parno and A. Perrig. Challenges in securing vehicular networks. In *Workshop on hot topics in networks (HotNets-IV)*, 2005.

[26] M. Raya and J. P. Hubaux. Securing vehicular ad hoc networks. *Journal of Computer Security*, 15(1):39–68, 2007.

[27] M. Raya, P. Papadimitratos, I. Aad, D. Jungels, and J. P. Hubaux. Eviction of misbehaving and faulty nodes in vehicular networks. *IEEE Journal on Selected Areas in Communications*, 25(8):1557–1568, 2007.

[28] R. K. Schmidt, T. Leinmuller, E. Schoch, A. Held, and G. Schafer. Vehicle behavior analysis to enhance security in vanets. In *IEEE V2VCOM*, 2008.

[29] D. Schrank. *Urban Mobility Report (2004)*. DIANE Publishing, 2008.

[30] H. Stubing, J. Firl, and S. 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.

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

[32] G. Y. Yan, G. Choudhary, M. C. Weigle, and S. Olariu. Providing vanet security through active position detection. In *ACM VANET*, 2007.

[33] Y. Yang and R. Bagrodia. Evaluation of vanet-based advanced intelligent transportation systems. In *ACM VANET*, 2009.