AntibotV: A Multilevel Behaviour-Based Framework for Botnets Detection in Vehicular Networks

Rabah Rahal, Abdelaziz Amara Korba, Nacira Ghoualmi-Zine, Yacine Challal, Mohamed Ghamri-Doudane

AntibotV: A Multilevel Behaviour-based Framework for Botnets Detection in Vehicular Networks

Rabah Rahal*[a], Abdelaziz Amara-Korba[a], Nacira Ghoualmi-Zine[a], Yacine Challal[b] and Mohamed Yacine Ghamri-Doudane[c]

[a]Networks and Systems Laboratory (LRS), Badji Mokhtar-Annaba University, Annaba, Algeria
[b]Laboratory of Methods of Systems Design (LMCS), National School of Computer Science (ESI), Algiers, Algeria
[c]Laboratory of Informatics, Image and Interaction (L3i), University of La Rochelle, La Rochelle, France

ARTICLE INFO

Keywords:
ITS
Vehicular Networks
Botnet
Intrusion detection
Network Flow
Network Forensics

ABSTRACT

Connected cars offer safety and efficiency both for individuals as well as for fleets of vehicles, companies and public transportation. However, equipping vehicles with information and communication technologies also raises privacy and security concerns, which significantly threaten the user's data and life. Using a bot malware, a hacker may compromise a vehicle and control it remotely, for instance he can disable breaks or start the engine remotely. In this paper, besides in-vehicle attacks existing in the literature, we consider new zero-day bot malware attacks specific to vehicular context, WSMP-Flood and Geo-WSMP Flood. Then, we propose AntibotV, a multilevel behaviour-based framework for vehicular botnets detection in vehicular networks. The proposed framework combines two main modules for attacks detection, the first one monitors the vehicles activity at the network level, whereas the second one monitors the in-vehicle activity. The two intrusion detection modules have been trained on historical network and in-vehicle communication using decision tree algorithms. The experimental results showed that the proposed framework outperforms existing solutions, it achieves a detection rate higher than 97% and a false positive rate lower than 0.14%.

1. Introduction

With the proliferation of connected cars, the concept of vehicular networks has moved to another new level. Vehicular networks became a distributed transport fabric capable of making its own decisions about driving customers to their destinations [1]. Applications of vehicular networks vary from safety applications such as blind spot warning and traffic light violations to entertainment such as streaming media or convenience such as parking space identification. Recently established standards such as the dedicated short-range communication standard (DSRC) help achieve effective communications between vehicles and the infrastructure. DSRC is a communication technology that relies heavily on several cooperatives and interoperable standards: IEEE 802.11p, IEEE 1609.x, SAE J2735 Message Set Dictionary, and the emerging SAE J2945.1 standard. IEEE 802.11p and IEEE 1609.4 are used to describe the physical and the Media Access Control (MAC) layer of the system respectively. While IEEE 1609.3, with its two supported stacks (Internet Protocol version 6 (IPv6) stack for non-safety applications and WAVE Short Message Protocol (WSMP) stack for safety applications) is used to describe the network and transport layers. Security functions and services are described by an IEEE 1609.2 standard protocol. While SAE J2735 and SAE J2945.1 standards are used to define the format of the messages exchanged over the network

Vehicular networks can help to increase transportation safety and efficiency [2]. However, it raises privacy and security concerns, which significantly threaten the network operations and the user data. One of the most dangerous cybersecurity threats is when the on board computer of a connected vehicle get compromised and exploited by a remote attacker. This cyberthreat is known as vehicular bot malware. Unlike the other security threats, it can be used to execute remotely multilevel malicious tasks: (1) distributed network attacks (DDoS) [3]; (2) violating drivers personal data (location privacy [4] or illegal GPS tracking and eavesdrop drivers and passengers' conversations) [5]; (3) controlling the bot vehicles remotely (opening the door, starting the engine, turning on the lights, driving the vehicle away or disabling breaks); (4) misleading the driver by giving him wrong information about the vehicle state (falsifying the fuel level, changing the speedometer reading and displaying failure information on the instrument panel cluster) [6].

Despite the harmful impact of vehicular bot malwares on the driver's privacy and safety, there are few researches [7, 8], that have considered this issue. Garip et al. [8] proposed SHIELDNET, a machine learning based botnets detection mechanism. As botnet they considered GHOST [7], a botnet communication protocol that uses Basic Safety Messages BSMs to dissimulate its communication over the control channel. SHIELDNET detects the use of GHOST, and identifies vehicular botnet communication, by looking for abnormal values of specific BSM fields, which are messages used by security applications only in vehicular networks and cannot be found in other types of networks. Although its

*This research is a result from PRFU project C00L07UN23 0120180009 funded in Algeria by La Direction Générale de la Recherche Scientifique et du Développement Technologique (DGRSDT).

rabah.rahal@univ-annaba.org (R. Rahal*);
abdelaziz.amara.korba@univ-annaba.org (A. Amara-Korba);
ghoualmi@yahoo.fr (N. Ghoualmi-Zine); y_challal@esi.dz (. Challal);
yacine.ghamri@univ-lr.fr (M.Y. Ghamri-Doudane)
ORCID(s):

effectiveness, SHIELDNET relies on a specific communication protocol (Ghost), thus it would not be effective if the bot master changes the communication protocol.

In this paper, we propose a machine-learning botnet detection approach that does not suppose a particular botnet communication protocol. As vehicular bot malware is a cyber threat operating within in-vehicle and network communication levels, we propose a multilevel behaviour-based framework for botnets detection in vehicular networks (AntibotV). The proposed framework monitors the vehicle's interaction with the outside by analyzing the network traffic. It also monitors the in-vehicle activity to detect suspicious operations that may relate to bot malware activity. In addition, this paper considers new zero-day bot malware attacks that could be carried out exclusively against vehicular networks communication stack: 1) wave short message protocol flood (WSMP flood); 2) and geographic wave short message protocol-Flood (Geo WSMP flood). Both attacks target the wave short message protocol, which is used in vehicular networks for safety and convenience packets transfer.

The contribution of this paper is two-fold:

(i) First, we identify a set of zero-day attacks that can be executed by a hacker through compromising a vehicles on-board computer using bot malwares. We provide a detailed description about two zero-day DDoS attacks, and information theft attacks specific to vehicular context.

(ii) Second, we propose a multilevel botnet detection framework based on decision trees to detect zero-day and existing attacks by monitoring the vehicles activity at the network and in-vehicle level.

The rest of the paper is organized as follows. Section 2 provides a background related to vehicular networks architecture and botnets communication. Section 3 summarizes the existing works in the literature related to botnet detection. In section 4 we describe the threat models. Section 5, describes in detail the proposed framework, AntibotV. In section 6, we provide detailed description of the dataset generation steps and discuss the obtained results. Section 7 concludes the paper and draws some line of future work.

2. Background

This section provides a description of network and in-Vehicle architecture and communication, as well as a brief state of the art about botnets detection.

2.1. Vehicular Networks Architecture

A vehicular network organizes and connects vehicles with each other, and with mobile and fixed-locations resources (Road Side Units). Many telematics architectures, including navigation services, traffic information, location-based services, entertainment services, emergency and safety services have been provided. In these architectures, traffic information and navigation services are generally provided by central TSPs (Telematics Service Providers) through a vehicle-to-infrastructure communication [9]. On the other hand, the emergency and safety services are supplied by an Onboard Unit (OBU) installed by the individual car manufacturers, in ordre to allow the mutual communication among different vehicular nodes (vehicle-to-vehicle) [10].

2.1.1. Network stack architecture

In order to ensure vehicle-to-vehicle and vehicle-to-infrastructure communication, the automotive industry has developed the dedicated short-range communication standard (DSRC) (Figure 1). A communication technology that relies heavily on several cooperatives and interoperable standards [11]. At the PHY and MAC layers, DSRC utilizes IEEE 802.11p Wireless Access for Vehicular Environments (WAVE). WAVE is an approved amendment of the IEEE 802.11 standard that enable secure wireless communication and physical access for high speed (up to 27 MB/s), short range (up to 1000 m), and low latency. The spectrum allocated to it is from 5.850 to 5.925 GHz, divided into seven 10 MHz channels [12]. Channel 178 is reserved for control information and the others six channels are used for service applications (Figure 2).

At the MAC sublayer extension of DSRC, the IEEE 1609.4 standard is deployed. It is used for priority access, management services, channel switching and routing [13] in order to enable to operating efficiently on seven channels alternately [14]. As regards the IEEE 1609.2 standard, it includes techniques used to secure application messages and describe administrative functions necessary to support the core security functions [15].

At the Network Layer, DSRC uses the IEEE 1609.3 standard. A standard that supports two protocol stacks, Wave Short Message Protocol (WSMP) and IPv6. The choice between using WSMP or IPv6+UDP/TCP depends on the requirements of the application. For the ones that depend on the routing capabilities to transmit multi-hop packets like commercial applications, IPv6, UDP and TCP are used. However, the applications that require the transfer of single-hop messages like security and convenience applications, WSMP protocol is used. Unlike the IP, UDP and TCP protocols, the WSMP is a WAVE network layer unique protocol and can be found only in a vehicular networks. It is used only to support high priority and time sensitive communication.

Finally, for the format of the messages exchanged over DSRC, like data frames and elements used by the applications, they are defined in the SAE J2735 and SAE J2945.1 standards. SAE J2735 represents a dataset that contains message syntax. It contains many types, we cite among them the Basic Safety Message (BSM) (periodically transmitted to provide current information and status) and the Common Safety Request (CSR) [16]. Other messages norms for the V2V safety applications are specified with the SAE J2945.1 standard.

2.1.2. In-Vehicle architecture

Nowadays, we see that the automotive industry is converging to replace the mechanical components of the vehicle with other electronic components labelled electronic control units (ECUs) (expected to reach 3.29 billion deployed

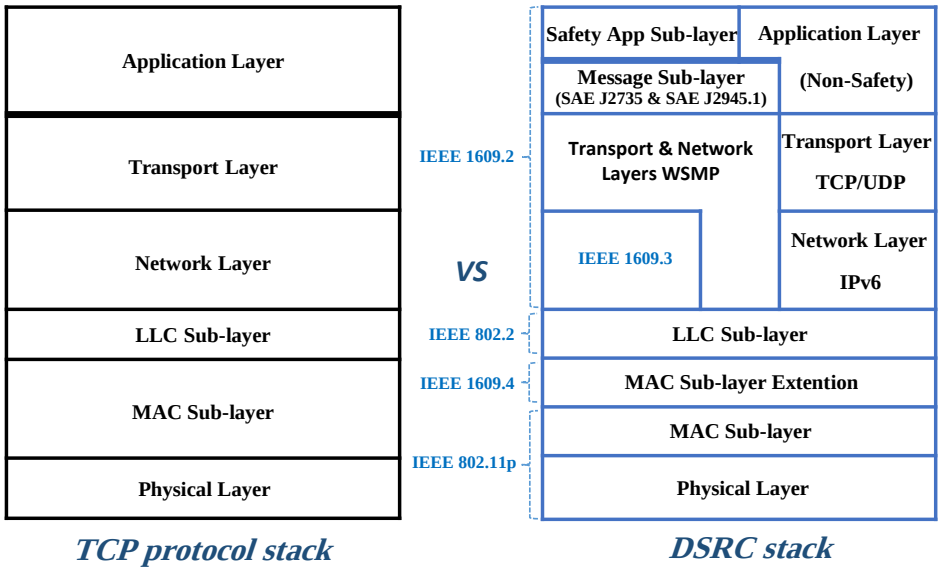

Figure 1: TCP vs DSRC stack

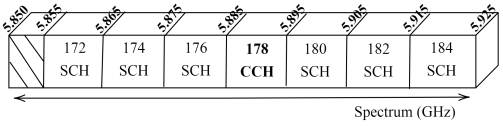

Figure 2: WAVE radio channels

units by 2025 [17]). ECUs simplify the interior architecture of vehicles, thus repair and diagnose even for those who know nothing about vehicles. Each ECU contains its sensors and actuators, it gets input from its sensors and implements specific functions by its actuators. Communication between these ECUs is ensured through a dedicated bus type especially for vehicular networks; called Controller Area Network (CAN bus). ECUs and CAN buses together form the In-vehicle network (Figure 3).

In this In-vehicle network, there are two types of CAN bus (high-speed and low-speed) connected by a gateway [6]. For the communication of critical modules (power train, brake, etc.), the high-speed CAN bus is used. For the other types of modules (telematics, body control, etc.) we use low-speed CAN bus. The transmission on CAN bus is done sequentially. However, if more than one device transmits at the same time, a media access control (MAC) protocol Carrier-sense multiple access with Collision Resolution (CSMA-CR) is used. CSMA/CR uses priorities in the frame header to avoid collisions [18].

2.2. Botnets

A botnet is a collection of internet-connected devices (computers, smartphones, IP cameras, routers, IoT equipements, ..) called bots, and which were infected by a malware, in order to be controlled remotely by an operator (botmaster) without users' knowledge. The communication between the botmaster and the bots is ensured through the Command and Control (C&C) server. Botnets can be used to achieve harmful attacks, such as: launching Distributed Denial-of-Service (DDoS) on rival websites or services, send spam, distribute malwares, stole user/equipment private information, and applying interior activities on the infected devices. For example, the Mirai botnet [19] in 2016, was able to carry out a massive DDoS attack that brought down major sites like Amazon, Netflix, Paypal, and Reddit [20].

3. Related Work

Researchers have worked on detecting botnets and to overcome their negative impact. They proposed methods suited to the characteristics of each type of network (communication stack, protocols, characteristic of equipment's, etc). Other worked on the protection against DDoS attacks that can be caused by a botnet [21, 22, 23, 24]. We find also different anomaly detection, revolutionary and hybrid classification techniques proposed to deal with new types of botnets [25, 26, 27, 28].

In [21], the authors proposed an ML-based DDoS detection and identification approach using native cloud telemetry macroscopic monitoring. A lightweight method and completely agnostic to specific protocols and services, which can detect any kind of DDoS attack that target the resources without the need for previous training. The authors in [22] have worked on the detection of Low-Rate DDoS (LR-DDoS) attack (exactly the Shrew attack). They proposed new mechanism which not only detects and mitigates the shrew attack but traces back the location of the attack sources as well. The attack is detected using the information entropy variations, and the attack sources are traced-back using the deterministic packet marking scheme. If the DDoS attack is caused by a botnet, the traceability mechanism can be used to identify

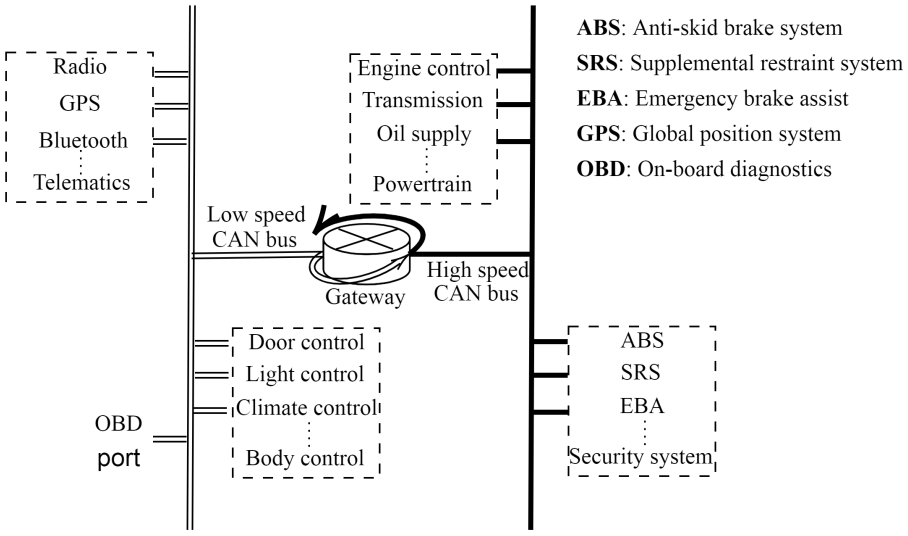

Figure 3: In-vehicle network

bot nodes in the network. Approaches that deal with DDoS can be used to mitigate the effects of botnets (and even to identify botnodes), however, if the botnet is used for other attacks (e.g. information theft), DDoS detection techniques will not be effective.

In [25], the authors proposed a model (known as AS-IDS) that combines two detection approaches (anomaly-based and signature-based) to detect known and unknown attacks in IoT networks. The proposed model has three phases: traffic filtering, preprocessing and the hybrid IDS. The signature-based IDS subsystem investigates packets by matching the signatures, and categorized them as intruder, normal or unknown. The anomaly-based IDS subsystem employs Deep Q-learning to identify unknown attacks. In [29], The proposed method attempts to identify those mallicious Botnet traffic from regular traffic using novel deep learning approaches, exactly Gatted Recurrent Units (GRU) model. In a second step, they introduced adjusted hyper parameters to make the computational complexity low and more accurate than the existing models. However, the use of two detection approaches could cause an unbearable overload, which does not suit the type of network whose nodes have limited computing and storage capacities.

In the case of traditional computer based botnet such as Ebury botnet [19], network-based detection is one of the most effective detection technique that has attracted the attention of many researchers [30, 31, 32, 33, 34]. By analyzing the rate of failed connection and network flow features, network-based detection technique identifies the exchanged traffic between the C&C server and the bots.

In regard to mobile networks such as WireX botnet [19], it is the behavioral analysis methods that have caught the attention of researchers [35, 36, 37]. Different level in the Android OS stack gives completely different set of behavioural data. Behaviour-based detection method aims to detect malicious application by monitoring different layer of Android OS (API calls, ...). We find also other researchers who have used traditional detection methods such as rules-based [38], and signature-based [39].

Recent researches [40, 41, 42] on IoT botnets detection used unsupervised anomaly detection techniques in order to detect unseen botnet attacks. Goebel et al. [43] proposed a signature-based method to minimize false positive rate. Also detection approaches using DNS related network traffic method have been proposed [44, 45, 46, 47, 48].

The existing botnets detection mechanisms cannot be applied directly for vehicular networks due to differences in terms of communication stack, protocols, frames format and architecture. In the context of vehicular networks, to the best of our knowledge, there is only one research [7] that has tackled the problem of botnets detection. Garip et al. [7] focused on the communication protocol between the botmaster and the bot vehicles. They investigated the usage of periodic basic safety message to transmit commands from the bot master to the bot vehicles. This communication protocol called "Ghost" allows the hacker to hide its remote communication with the infected vehicle. The authors considered two bot malware attacks feasible against vehicular networks: Botveillance [49]; and Congestion attacks [50]. Botveillance is an adaptive cooperative surveillance attack against pseudonymous systems in vehicular networks, which is based on vehicular botnet and performed by vehicles themselves without depending on any additional hardware. It is used to track vehicles of interest or violate the privacy of drivers. On the other hand, the congestion attack is based on the vehicular congestion avoidance application. It uses bot vehicles to spread wrong congestion information to the other legitimate vehicles, in order to cause congestion on a specific roads or areas.

The same authors [8] proposed SHIELDNET, a vehicular botnets detection mechanism that uses machine learning techniques to detect GHOST usage and identify the bot-

net communication. SHIELDNET detects botnet activity by looking for anomalous values of specific BSM fields. Although the efficiency of the proposed solution, the Ghost protocol is a single hop range, which makes the command of the botmaster sent only to its neighbours, making the range of communication very short and unreachable by far bot vehicles. Moreover, if the botnet changes the communication protocol, it will not be effective. Furthermore, the scenario of botnet in vehicular networks is not real. Because transmitting commands through V2V communication will make the botmaster control capacity very limited and it need to be on the road and near from other botnet nodes. Therefore, the ideal way to imagine a botnet in vehicular networks is where the botmaster is anywhere, and can control those vehicles using V2I communication and send commands through infrastructure. On the other hand, if the installed malware is used to apply in-vehicular activities, SHIELDNET will not be able to detect any abnormal communication or behaviour.

To the best of our knowledge, no research on botnets in vehicular networks has taken into account activities at the network and in-vehicle levels together. In this paper, our research aims to implement a botnet detection approach that does not suppose a particular botnet communication protocol. A multilevel behavior-based framework (AntibotV) for botnets detection in vehicular networks that monitors the vehicle's interaction with the outside by analyzing the network traffic. It also monitors the in-vehicle activity to detect suspicious operations that may relate to bot malware activity.

4. Threat Models

In this section, we provide a detailed description of the three categories of cyberattacks that can be executed against a target vehicle using bot malwares. First, we provide a detailed description about two zero-day DDoS attacks. Then based on existing attacks on privacy, we consider new scenarios applicable to the vehicular context. Finally, we present some in-vehicle attacks that exist in the literature. Figure 4 shows the different categories of cyberattacks that a hacker may execute using a bot malware.

4.1. DDoS attacks

Due to the differences between DSRC stack and the TCPIP stack, it is important to consider DDoS attack scenarios specific to the vehicle network, and not be limited to DDoS attacks common to all IP networks. Therefore, in this paper, we consider two zero-day attacks specific to vehicle networks. Both attacks exploit the WAVE Short Message Protocol (WSMP) which is used by security and traffic management applications for the transfer of critical data such as: vehicle speed, kinematic state, etc. Both attacks can prevent the transfer of safety messages between vehicles and thus cause catastrophic damage.

The WSM packets (Figure 5) exchanged between vehicles are composed of the following fields: WSMP version that gives the version of the protocol, channel number and data rate to specify which channel and data rate are used for the transmission, WAVE element ID represents the WSMP header, WAVE Length to specify the length of the packet, and the WSM Data field contains the payload data. The Provider Service Identifier field (PSID), identifies the service that the WSM payload is associated with. For example, if an application tries to get access to the WAVE service, it should be registered with its unique Provider Service Identifier (PSID). The WAVE provider devices use PSID in its announcement messages to indicate that a certain application is provided by this device. On the other hands, as the vehicle passing the road side device, user devices, which may host such application, upon reception of such announcement, compares to check if there is a match between the PSIDs in the announcements and PSIDs in its own tables, then the vehicle establish communication with that road side unit [11, 12].

In both attacks the hacker attempts to misuse the WSMP protocol. In the first attack, WSMP flood, the hacker sends to the victim vehicle a WSM packets with unknown PSID field values (not associated to any WAVE service). Upon the reception of the forged WSM packet, the target vehicle attempts to check (lookup) for the corresponding entry within its PSID/Service table. Checking the PSID of one WSMP frames will not be a problem. However, checking the PSID of huge amount of WSM packets at the same time will exhaust the resources of the target vehicle and make it unable to respond or receive legitimate security and convenience applications packets.

The second attack, geographic WSMP flood, has the same operating mode as WSMP flood, but wider impact. The hacker broadcasts the forged WSMP messages to all its neighbours within a specific geographic area. Thus, it exhausts the resources of several vehicles, and it consumes the bandwidth of an entire geographic area.Before validating AntibotV with the WSMP flood attack, we tested it, and we found that it can decrease the throughput with 63% (the number of security and convenience frames arrived), which make the attack very effective (as shown in figure 6). Regarding the implementation of the scenario, the simulators used, as well as the details of the simulation (number of nodes, time of simulation, MAP, traffic collection, etc.), all these details are discussed in the subsection 6.1.1.

4.2. Theft of information

A team of academics from the University of California at San Diego and the University of Washington have found out that the audio surveillance inside the vehicles is possible [51]. In such situation, the attacker indirectly uses virtual assistance tools (like Siri [52]) by giving it malicious commands hidden in recorded music, innocuous-sounding speech [53] or a low-powered laser [54]. Siri begins recording the conversations using the vehicle's internal microphones, and sends them to a remote server belonging to the hacker every 10 or 20 seconds[55].

In the case of GPS tracking, the hacker may attempt to track the vehicle's location on real time, check the location or retrieve the complete trajectory. For the real time tracking, the GPS information must be sent every second (streaming)

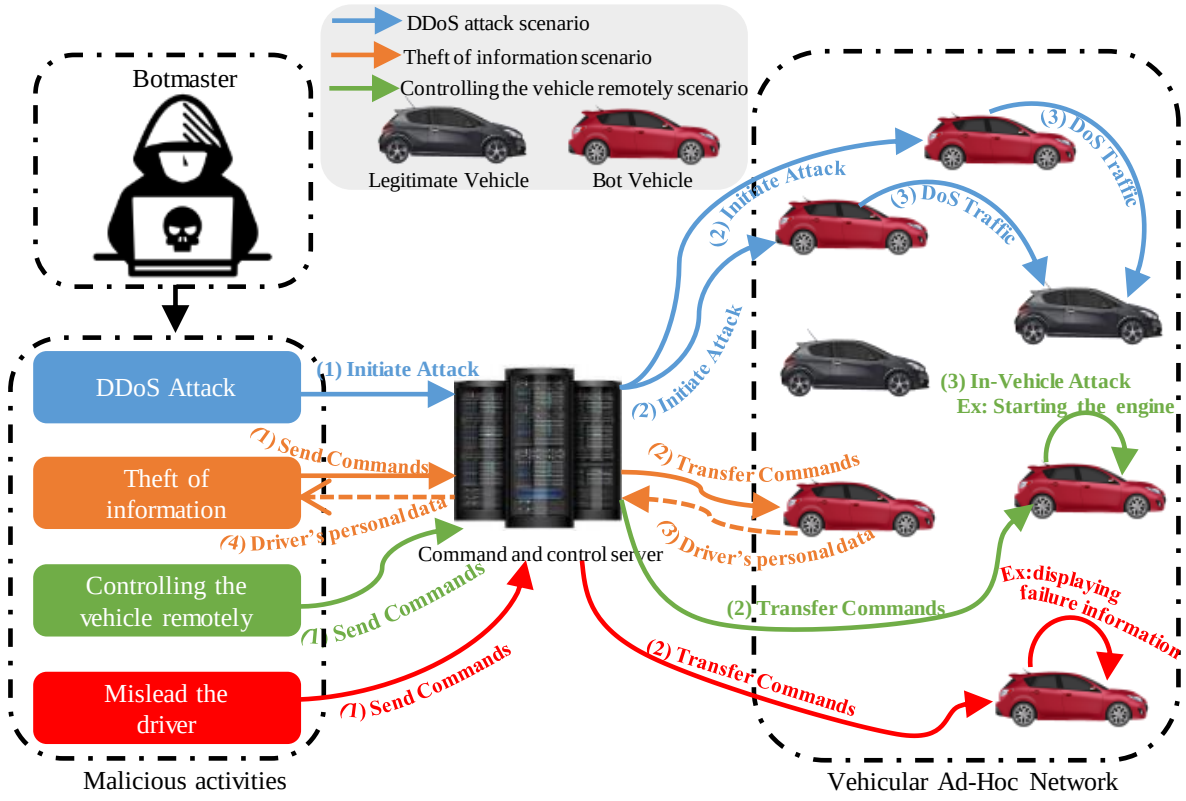

Figure 4: Vehicular Botnets

Table 1
Threat models

Category	Attack	Objective
DDoS	WSMP Flood	Exhausting the resources of the target vehicle and make it unable to respond or receive legitimate security and convenience applications packets.
	Geo-WSMP Flood	Exhausting the resources of several vehicles, and consuming the bandwidth of an entire geographic area.
Theft of information	GPS Tracking	Tracking the vehicles location on real time.
	Audio Surveillance	Recording the conversations using the vehicles internal microphones, and sends them to the botmaster.
In-vehicle	Frame Falsifying	Fabricate fake frames that contain erroneous data to mislead the driver: falsifying the fuel level, changing the speedometer reading
	Replay Attack	Replaying captured frames. Could be used to: opening the door, starting the engine
	DoS Attack	Monopolize the transmission channel to delay or prevent the transmission of legitimate CAN frames. Example: disable the brakes.

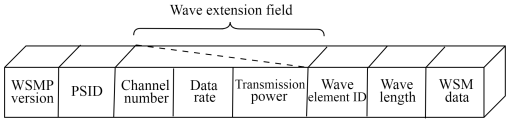

Figure 5: WSM frame

to the botmaster. For the other two types of GPS tracking, the information are sent periodically or on-demand. In this paper, we consider the real time GPS tracking scenario. The bot vehicle sends the latitude (4 bytes), longitude (4 bytes), speed (2 bytes), time (2 bytes) and direction coordination in streaming to the botmaster [56].

4.3. In-Vehicle attacks

The in-vehicle networks brought convenience to manufacturers, drivers and after-sales services. However, they raise vulnerabilities [6] that can be exploited by a hacker to conduct the following attacks:

1. **Frame Falsifying and injection**: CAN frames are sent in plaintext, which allows the bot malware within

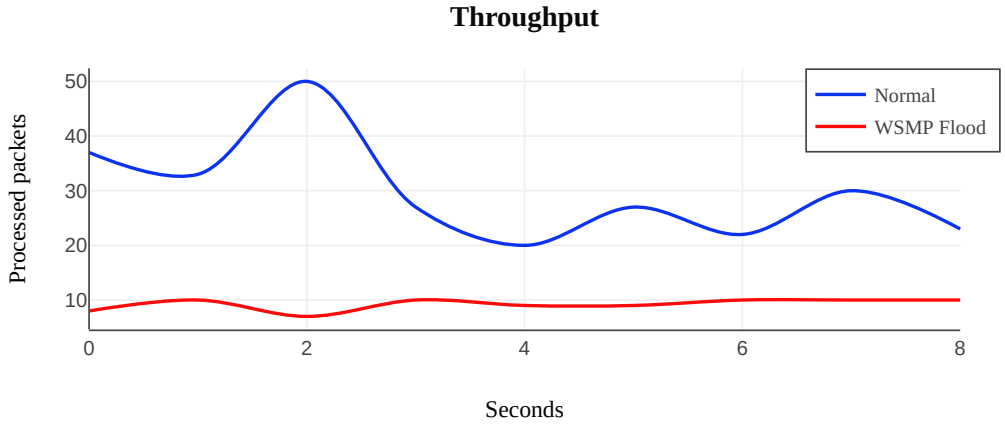

Figure 6: Normal throughput vs WSMP Flood throughput

the vehicle to retrieve and analyze their contents. Using the captured data, the bot malware can fabricate fake frames that contain erroneous data to mislead the ECUs. Subsequently, the fabricated frames are injected in CAN bus to carry out malicious activities such as: falsifying the fuel level, changing the speedometer reading or displaying failure information that may mislead the driver.

2. **Replay Attack**: The frames are transmitted through CAN bus using broadcast and without authentication. Thus, CAN frames can be easily captured by the bot malware to be analyzed and replayed in a second step. In [57], Koscher et al. found that the range of valid CAN frames is small. Hence, by iteratively testing CAN frames (i.e., the fuzzing test mentioned in [57]), adversaries are able to discover many functions of selected ECUs. The replay attack could be used to open the door, start the engine, turn on the lights or remotely drive the vehicle.

3. **DoS Attack**: messages transmitted with the smallest identifier are the messages with the highest priority. The bot malware could use this vulnerability to monopolize the transmission channel, and thus delay or prevent the transmission of legitimate CAN frames. For instance the bot malware can prevent a CAN frame from being transmitted to the ECU of brakes, which could lead to accident [6].

Table 1 provides a summary of the different security threats discussed earlier.

5. AntibotV Framework

In this section, we describe in detail the AntibotV, a multi-level framework to detect vehicular botnets. Moreover, we describe the detection process based on machine-learning algorithms.

5.1. AntibotV overview

We suppose that a network traffic generated by a bot malware is different from legitimate traffic, in other words the bot malware alters the traffic pattern of the vehicle. This hypothesis comes from the fact that connected cars run specialized applications related to safety and convenience, such as: cooperative collision warning, V2V post crash notification, congested road notification, etc... Running specialized and particular applications, makes connected vehicles' network traffic pattern regular as long as it is not compromised. Thus, we believe bot malware that compromises a connected vehicle should alter its network traffic pattern. Likewise, at the in-Vehicle level, because of the regularity of communication pattern, it is possible to identify a legitimate in-Vehicle traffic pattern, and thus, detect malicious bot activity.

The AntibotV is a host-based intrusion detection system, running the system in a vehicle, rather than at some point of the network. The proposed system monitors communication with outside through analyzing network traffic, and it monitors in-vehicle communication by analyzing CAN bus frames. The two-level monitoring allows an effective detection of bot malware activity which consists in sending data and receiving commands from bot master, at network level, and executing control commands, falsify the fuel level, change the speedometer reading or display failure information at in-vehicle level.

The network traffic is characterized using a set of statistical features such as flow duration, total packets sent in forward direction, minimum packet length, etc (detailed description of used features is provided in the next section). To characterize the in-vehicle communication a set of twelve features is used including timestamp (recorded time), CAN ID (identifier of CAN message in HEX), DLC (number of data bytes, from 0 to 8), DATA[0 7] (data value in byte).

Regarding network traffic, it is independent of the type of vehicle, model, or year and it depends only on the protocol stack used, which implies that AntibotV is applicable in all vehicles that use DSRC standard as wireless access technol-

Table 2
List of network features

Features		Description
Nb	**Name**	
1	Flow Duration	Duration of the flow in microsecond
2	Flow IAT Mean	Mean time between two packets sent in the flow
3	Flow IAT Std	Standard deviation time between two packets sent in the flow
4	Flow IAT Max	Maximum time between two packets sent in the flow
5	Flow IAT Min	Minimum time between two packets sent in the flow
6	Fwd IAT Tot	Total time between two packets sent in the forward direction
7	Fwd IAT Mean	Mean time between two packets sent in the forward direction
8	Fwd IAT Std	Standard deviation time between two packets sent in the forward direction
9	Fwd IAT Max	Maximum time between two packets sent in the forward direction
10	Fwd IAT Min	Minimum time between two packets sent in the forward direction
11	Down/Up Ratio	Download and upload ratio
12	Idle Mean	Mean time a flow was idle before becoming active
13	Idle Std	Standard deviation time a flow was idle before becoming active
14	Idle Min	Minimum time a flow was idle before becoming active
15	Tot Fwd Pkts	Total packets in the forward direction
16	Flow Pkts/s	Number of flow bytes per second
17	Fwd Pkts/s	Number of forward packets per second
18	Bwd Pkts/s	Number of backward packets per second
19	total_Bwd_Packets	The total number of packet in the bwd direction.
20	total/min/max/mean/std_f/b_Pktl	The size of packets and the standard deviation size in fwd or bwd direction.
21	Avg_Packet_Size	The mean packets size.
22	f/b_AvgSegmentSize	The median noticed size in the fwd or bwd direction.
23	f/b_AvgPacketsPerBulk	The mean number of packets bulk rate in the fwd or bwd direction.
24	f/b_AvgBulkRate	The average number of bulk rate in the fwd or bwd direction
25	act_data_pkt_forward	The number of packets with at least 1 byte of TCP data payload in the fwd direction
26	min_seg_size_forward	The smallest segment size noticed in the fwd direction.
27	Total_f/b_headers	The total number of bytes used for headers in fwd or bwd direction
28	f/b_AvgBytesPerBulk	The mean number of bytes bulk rate in the fwd or bwd direction.
29	Init_Win_bytes_forward/backward	The total number of bytes sent in initial window in the fwd or bwd direction.
30	total/min/max/mean/std_Bwd_iat	The total, max, min, mean, and standard time between packets for the bwd direction.
31	f/b_psh/urg_cnt	The number of the PSH or URG flags were set in packets in the fwd or bwd direction.
32	Min/mean/max/std_active	The min, max, mean, and std time a flow was active before becoming idle.
33	Flow_Byts_PerSecond	The number of a flow bytes per second.
34	min/max_flowpktl	The length (min, max) of flow.
35	Flow_fin/syn/rst/psh/ack/urg/cwr/ece	The number of packets with flags.
36	Sflow_f/b_Packet	The average number of packets in a sub flow for the fwd or bwd direction.
37	Sflow_f/b_Bytes	The average number of bytes in a sub flow for the fwd or bwd direction.

ogy. Concerning CAN bus standard, it consists of two versions based on the length of the arbitration field. CAN bus 2.0A defines an 11-bit standard frame format while CAN bus 2.0B is compatible with data messages in a standard frame and extended frame format. Our framework mainly focuses on CAN ID, DLC and the DATA fields (Table [3]), and the three fields are found in both versions (2.0A and 2.0B) of CAN bus protocol and they have the same length. This implies that AntibotV is a general framework for all vehicles that uses the standard DSRC as wireless access technology and CAN bus protocol with its two versions, regardless the model of the vehicle.

Being a host-based system, AntibotV should not consume a large amount of the vehicles resources, because that would impact its other operations. For that reason, we move the computational load of training classification model to a centralised server on the cloud. Thus, the vehicle runs an already trained classification model, which would minimize the resources consumption of vehicle. Additionally, unlike signature-based models, AntibotV is a behavior-based model based on machine learning algorithms, which means there is no overhead to maintain a signature database up to date, nor huge computational resources like deep learning-based models. In addition, it does not require any hardware modification.

AntibotV has a modular architecture inspired by the IDMEF (Intrusion Detection Message Exchange Format) architecture proposed by the IDWG group [58]. The architecture is mainly composed of three modules: traffic collection module, analyzer module, and manager module. The first module collects network traffic and in-Vehicle CAN frames. The analyzer module is responsible for analysing the vehi-

cles traffic data. The manager module handles alerts, send notifications, and updates classification model for both traffic. Detailed description of these modules is provided in the next sections.

5.2. Traffic collection module

Used to collect and process the vehicle traffic and apply several pre-treatment operations in order to extract from the raw data a vector of features, which will be used by the analyzer module. It collects two types of traffic:network flow and CAN bus frames.

5.2.1. Network traffic collector

This module collects information about network traffic form exchanged packets. Each packet is mapped to a network flow identified by five attributes namely source IP, destination IP, source Port, destination port and Protocol. The RFC 3697 [59] defines traffic flow as "a sequence of packets sent from a particular source to a particular unicast, anycast, or multicast destination that the source desires to label as a flow. TCP flows are usually terminated upon connection teardown (by FIN packet) while UDP flows are terminated by a flow timeout. Table 2 shows the list of network flow-based features extracted for each network flow. At the end, the network traffic collector generates a vector of calculated features, then transfers it to the analyzer module.

5.2.2. In-vehicle traffic collector

The in-vehicule traffic collector module analyses frames exchanged on CAN bus. Unlike the network traffic, which is processed as flows, the in-vehicle traffic is analysed using deep frames inspection technique. The analysis is carried out through real time observation of frames as they traverse CAN bus links. A CAN bus frame contains the following fields: the Start of Frame (1b), Message ID (an 11b identifier that represents the message priority), Control fields (3b), Data Length (number of data bytes, 4b, from 0 to 8), Data[0 7] (Data to be transmitted, 0-64b), CRC (15b), ACK fields (3b), End of Frame Delimiter (7b) [60].

The in-vehicule traffic collector generates a features vector with the following fields: timestamp (recorded time), CAN ID, DLC and the DATA, as illustrated in table 3. The timestamp and CAN ID fields could be used to detect DoS and replay attacks, which exploits the vulnerability of ID priority as described previously in section 4. The DLC and DATA fields could be used to detect falsified and injected frames. Then, the feature vector is transferred to the analyzer module.

Both network and in-vehicle features vectors are pre processed before their transfer to the analyzer module. Preprocessing operations concern mainly removing missing values and scaling feature values using z-transformation. The pseudo codes algorithm 1 summarizes the different steps of the traffic collection module.

Table 3
CAN bus frames feature vector attributes

Features	Description
Timestamp	Recorded time (s)
CAN ID	Identifier of CAN message
DLC	Number of data bytes, from 0 to 8
DATA[0]	Data value (byte)
DATA[1]	
DATA[2]	
DATA[3]	
DATA[4]	
DATA[5]	
DATA[6]	
DATA[7]	

Algorithm 1: Traffic Collection Module

```
BEGIN
Input : CBF (CAN bus frame), NF (Network flow)
Variables :: NFw (Network flow feature vector),
  CBFw (CAN bus frame feature vector)
if (NF is captured) then
    Extract Features from (NF);
    Generate (NFw);
    Preprocess (NFw);
    Send (NFw) to Analyzer Module;
end if
if (CBF is captured) then
    Extraction of (Timestamp, CAN ID, DLC, DATA)
      from CBF;
    Generate (CBFw);
    Preprocess (CBFw);
    Send (CBFw) to Analyzer Module;
end if
END
```

5.3. Analyzer module

It is the most important module of the framework. Since this module handles two types of independent traffic, it uses two analyzers. The first analyzer is responsible of analyzing network traffic, the second one to analyze CAN bus frames. The network analyzer is a classifier trained using supervised machine learning algorithms on legitimate and malicious network traffic. The legitimate network traffic is generated by running specialized vehicular application related to safety [61], convenience [62] and commercial [63] applications. The malicious network traffic is related to typical bot malware's activities such as DOS and information theft attacks. The in-Vehicle analyzer is a classifier trained using supervised machine learning algorithms on legitimate and malicious (DoS, Fuzzy, RPM and Gear) CAN frames. Both classifiers are generated in a central server then integrated into the framework. Within the vehicle, the two classifiers are used for continuous monitoring of network traffic and CAN bus traffic.

The network analyzer uses the calculated network features to classify the received network flow in one of three classes: normal, DOS, or information theft. The in-Vehicle analyzer classifies CAN bus frames based on the calculated features into four classes: normal, DOS, frames injection or Replay attack. If a malicious network flow or CAN bus

frame is detected, then the analyzer module sends an alert to the manager module to take an immediate action. Otherwise, it will be ignored.

To train both analyzers we use the following supervised machine learning algorithms. For each analyzer we choose the algorithm that gives the best performances. The following algorithms are selected for their known efficiency and classification performances:

1. **Naive Bayes Algorithm:** It is a probabilistic classifier that makes classifications using the maximum a posteriori decision rule in a Bayesian setting. It operates on a strong independence assumption, which means that the probability of one attribute does not affect the probability of the other.

$$P(c \mid x) = \frac{P(x \mid c) * P(c)}{P(x)} \tag{1}$$

$$P(c \mid x) = P(x1 \mid c) * P(x2 \mid c) * ... * P(xn \mid c) * P(c) \tag{2}$$

$P(c|x)$ is the posteriori probability of class (target) given predictor (attribute). $P(c)$ is the prior probability of class. $P(x|c)$ is the likelihood, which is the probability of predictor given class. $P(x)$ is the prior probability of predictor.

2. **Support Vector Machine(SVM):** Each data item is plotted as a point in n-dimensional space (where n is number of features) with the value of each feature being the value of a particular coordinate. Then, the classification technique is performed to differentiate the classes and define which one the data points belongs to.

3. **K-Nearest Neighbour:** the k-nearest neighbors algorithm (k-NN) is a non-parametric method used for classification and regression [64]. It works based on minimum distance from the query instance to the training samples to determine the K-nearest neighbours. After gathering the K nearest neighbours, it take simple majority of these K-nearest neighbours to be the prediction of the query instance. Formally:

$$score(D, C_i) = \sum_{D_j \in KNN_d} Sim(D, D_j) \wp(D_j, C_i) \tag{3}$$

Above, *KNN(d)* indicates the set of K-nearest neighbors of the query instance

$$D * \wp(Dj, Ci) \tag{4}$$

with respect to class *Ci*, that is:

$$\wp(Dj, Ci) = \begin{cases} 1, Dj \in Ci \\ -0, Dj \notin Ci \end{cases} \tag{5}$$

Sim(D, Dj) : represent the similarity score of each nearest neighbor document to the test document. Moreover, it is used as the weight of the classes of the neighbor document.

For test document *d*, it should be assigned the class that has the highest resulting weighted sum.

4. **Decision Trees:** to build a decision tree, in this paper we used the ID3 (Iterative Dichotomiser 3) algorithm that uses entropy and information gain as metrics. Entropy characterizes the impurity of an arbitrary collection of examples, it can be defined formally as follow:

$$H(s) = \sum_{c \in C} -P(c) * log_2 P(c) \tag{6}$$

where *S* is the current data set for which entropy is being calculated, *C* is the set of classes in *S*, and *P(c)* represents the proportion of elements in class c to the number of elements in set S.

5. **Random Forest:** like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our models prediction.

6. **Neural Networks:** is one of the most known machine learning algorithms. It works based on several layers in order to analyze the data. Each layer tries to detect patterns on the input data. When one of the patterns is detected, the next hidden layer is activated and so on [65]. In this paper, we use Multi layer perceptron (MLP), a supplement of feed forward artificial neural network. It consists of three types of layers: the input layer, output layer and hidden layer. The input layer receives the input signal to be processed. The required task such as prediction and classification is performed by the output layer. An arbitrary number of hidden layers that are placed in between the input and output layer are the true computational engine of the MLP [66].

The pseudo codes algorithm 2 summarizes the different steps of the analyzer module.

5.4. Manager module

The manager modules handles alerts and triggers adequate response measures according to the detected attacks. Whenever the analyzer module detects a botnet activity, it sends an alert to the manager module. The latter logs the traces of the corresponding event and notify the driver. If a DoS attack is detected the manager module terminates network session with the victim vehicle. In the case of theft of information, the manager saves logs and notifies the driver. If the driver does not approve the transfer, the connection to the destination address will be blocked. Otherwise, the flow will be ignored. At the in-vehicle level, when the analyzer detects a CAN frame as belonging to a botnet activity,

Algorithm 2: Analyzer Module

```
BEGIN
Input : CBFw (CAN bus frame feature vector), NFw (Network flow feature vector)
Output : CBFw_Decision (CAN bus frame feature vector Decision), NFw_Decision (Network flow feature vector Decision)
if (NFw is Received) then
    NFw_Decision <- Network_Analyzer_SubModule (NFw);
    Send (NFw, NFw_Decision) to Manager Module;
end if
if (CBFw is Received) then
    CBFw_Decision <- In-Vehicle_Analyzer_SubModule (CBFw);
    Send (CBFw, CBFw_Decision) to Manager Module;
end if
END
Function Network_Analyzer_SubModule(N):
    Local Variable : N_Prediction (Network Traffic Prediction)
    N_Prediction = Machine Learning Algorithm(N);
    return N_Prediction;
End Function
Function In-Vehicle_Analyzer_SubModule(F):
    Local Variable : F_Prediction (Frame Prediction)
    F_Prediction = Machine Learning Algorithm(F);
    return F_Prediction;
End Function
```

Algorithm 3: Manager module response measures

```
BEGIN
Input: CBFw (CAN bus frame feature vector), NFw (Network flow feature vector), CBFw_Decision (CAN bus frame feature vector Decision), NFw_Decision (Network flow feature vector Decision)
if (CBFw_Decision classified as malicious) then
    Send an alert to the driver about CBFw
    Ask driver's approval for system reset
end if
if (NFw_Decision classified as DoS attack) then
    Save detailed logs of NFw
    Send alert to the driver
    Terminate network session
    Ask driver's approval for system reset
else
    if (NFw_Decision classified as theft of information) then
        Save detailed logs of NFw
        Send an alert to the driver
        if (information theft confirmed) then
            Terminate network session
            Ask driver's approval for system reset
        else
            Ignore NFw ;
        end if
    end if
    Ignore NFw;
end if
END
```

it sends an alert to the manager, which will notify the driver. To avoid interrupting wrongly vehicle's services, whatever the type of detected attacks, the manager module asks for driver's approval before undertaking any response measures. When the driver is notified depends on the attack. The most serious cases (disabling brake attacks) require direct notification. However, in other scenarios that do not have an impact on the driver's life, the notification is not done while driving, but in the moments when the vehicle is completely stopped, in order to avoid unnecessary shocks. The pseudo codes algorithm 3 summarizes the different responses measures.

Terminating malicious processes running on the vehicle does not mean that it will not be executed again. To get rid of this inappropriate malware that was detected, the system of the vehicle must be reset. The resetting process removes the applications and files installed on the system, which might be the carriers of the malware. The manager module has to ask the permission of the driver before the reset operation.

Finally, in order to keep the trained model updated (in the face of new zero-day bot malware attacks), a new training session is initiated automatically whenever the accuracy of the model starts degrading. The manager module begins to send the incoming traffic collected by the collection module to a remote server on the cloud, which is used to perform the exhaustive calculation operations. A combination of the new collected data and old training data are used together to build and deploy the new model. In the first phase, the gathered new data sets are been analyzed. In the second phase, a correlation algorithm could be used to correlates the relationships between the old training data and would be able to predict the new collected malware variants data. In the final phase, the analyzer module uses the old and new data for training, so it can be able to detect the malware traffic and give appropriate output. The benefit of such update technique is that it is completely automated.

6. Experimentation

In this section, we provide the evaluation results of the proposed framework. First, we describe in details the datasets used in this research, and the pre-processing operations we have carried out. Since the proposed framework monitors the in-vehicule and network communication, we have used two datasets to evaluate its performance, the first one contains vehicular network traffic, the second one in-vehicule traffic [67]. A detailed description of the two datasets is provided below.

6.1. Network traffic dataset

6.1.1. Generating network traffic data

To the best of our knowledge, no real vehicular network traffic dataset including botnet traffic is publicly available. Thus, we simulate vehicular botnet activity discussed in sec-

Table 4
Vehicular Networks applications

Type	Abbreviation	Description
safety oriented	BSM	Basic Safety Message: a packet transmitted approximately 10 times per second. It contains vehicle's state information, like speed, GPS coordination, etc.
	CCW	Cooperative Collision Warning: an application that aims to help the driver to avoid collisions by giving him the kinematics status messages collected from the vehicles around.
	CVW	Cooperative Violation Warning: roadside units send to drivers the necessary information when approaching to a signal phase or a red light.
	EEBL	Emergency Electronic Brake Light: If a vehicle braking hard, it broadcasts a message to inform the other vehicles.
	PCN	V2V Post Crash Notification: If a vehicle is involved in a crash, it broadcasts a message to inform the other vehicles.
	RFN	Road Feature Notification: if a vehicle detects an advisory road feature (acute turn), it broadcasts a message to inform the other vehicles.
	RHCN	Road Hazard Condition Notification: if a vehicle detects a road hazard (rocks, animals, ice), it broadcasts a message to inform the other vehicles.
	SVA	Stopped or Slow Vehicle Advisor: If a vehicle reduces its speed or stops, it broadcasts a message to inform the other vehicles.
convenience oriented	CRN	Congested Road Notification: If a vehicle detects a collision, it sends a message to alert the other vehicles so that they can take another lane.
	PAN	Parking Availability Notification: this application is used to request the road side unit to provide the closest parking and if they contain free parking spaces.
	PSL	Parking Spot Locator: It is an exchange of information between a vehicle and the Parking Space Locator roadside unit. It is used to provide free parking places in a parking lot.
	TOLL	Free Flow Tolling: it permits to apply an e-payment while passing through a highway toll gates.
	TP	Traffic Probe: The kinematics status messages are collected and transmitted through road side units to the traffic management center.
commercial oriented	CMDD	Content, Map or Database Download: a vehicle connect to a wireless hot-spot to download content (maps, multimedia, or web pages).
	RTVR	Real-time Video Relay: a vehicle may initiate transfer of real-time video that may be useful to other drivers in the area.
	RVP/D	Remote Vehicle Personalization/Diagnostics: the drivers can connect to a hot-spot to download the latest personalized vehicle settings.
	SA	Service Announcement: Fast food or restaurant use an infrastructure that send periodically Hello messages to announce the vehicles of its presence.

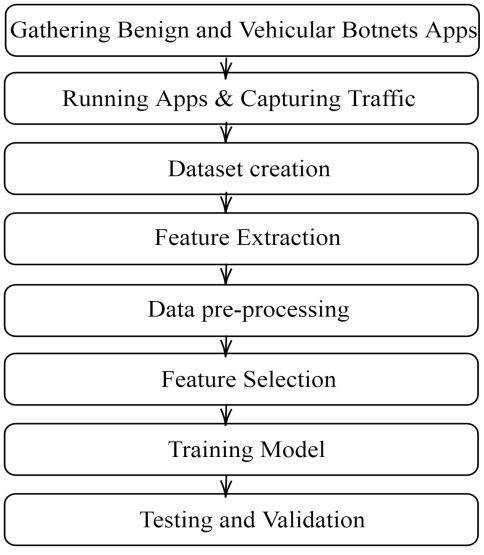

Figure 7: Dataset generation flow chart

tion 4. To generate realistic benign vehicular network traffic, we have implemented 17 applications including safety, convenience and commercial applications, the list of applications (inspired from [68]) with their brief description is provided in table 4, and the simulation parameters and scenarios are described further in this section. To ensure that the generated traffic is representative of real benign traffic and covers diverse types of application, we have considered the following factors for choosing applications: 1) physical-layer channel (CCH & SCH); 2) transfer protocols (IP & WSMP); 3) message TTL (single-hop & multi-hop); 4) routing protocol (geocast, broadcast & unicast); 5) trigger Condition (On-demand & event-triggered); 6) and communication technology (V2V & V2I). Table 5 provides a categorization of the benign applications based on the aforementioned factors.

To generate malicious network traffic related to a bot malware activity, we have simulated the following bot activity scenarios: 1) WSMP flood; 2) geo WSMP Flood; 3) GPS tracking; 4) and information theft, through eavesdropping on drivers and passenger's conversations.

To simulate the aforementioned benign vehicular network applications and the bot malware activity, we implemented different scenarios based on nodes ID. Four scenar-

Table 5
Vehicular networks applications classified based on network attributes

Application		Channel		Protocol		Message TTL		Routing Protocol			Trigger Condition			Participants		
		CCH	SSH	WSMP	IP	Multi-hop	Single-Hop	Geocast	Broadcast	Unicast	Beaconing	Event-triggred	On-demand	V2V	V2I	Internet
safety oriented	BSM	✓	✗	✓	✗	✗	✓	✗	✓	✗	✓	✗	✗	✓	✗	✗
	CCW	✓	✗	✓	✗	✗	✓	✗	✓	✗	✓	✗	✗	✓	✗	✗
	CVW	✓	✗	✓	✗	✗	✓	✗	✓	✗	✓	✗	✗	✗	✓	✗
	EEBL	✓	✗	✓	✗	✓	✗	✓	✗	✗	✗	✓	✗	✓	✗	✗
	PCN	✓	✗	✓	✗	✓	✗	✓	✗	✗	✗	✓	✗	✓	✗	✗
	RFN	✓	✗	✓	✗	✓	✗	✓	✗	✗	✗	✓	✗	✓	✗	✗
	RHCN	✓	✗	✓	✗	✓	✗	✓	✗	✗	✗	✓	✗	✓	✗	✗
	SVA	✓	✗	✓	✗	✓	✗	✓	✗	✗	✗	✓	✗	✓	✗	✗
convenience oriented	CRN	✗	✓	✓	✗	✓	✗	✓	✗	✗	✗	✓	✗	✓	✗	✗
	PAN	✗	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	✓	✗	✓	✗
	PSL	✗	✓	✓	✗	✗	✓	✗	✗	✓	✗	✗	✓	✗	✓	✗
	TOLL	✗	✓	✓	✗	✗	✓	✗	✗	✓	✗	✓	✗	✗	✓	✗
	TP	✗	✓	✓	✗	✓	✗	✗	✗	✓	✗	✓	✗	✗	✓	✗
commercial oriented	CMDD	✗	✓	✗	✓	✗	✓	✗	✗	✓	✗	✗	✓	✗	✗	✓
	RTVR	✗	✓	✗	✓	✓	✗	✗	✗	✓	✗	✗	✓	✗	✗	✓
	RVP/D	✗	✓	✗	✓	✗	✓	✗	✗	✓	✗	✗	✓	✗	✓	✗
	SA	✗	✓	✗	✓	✗	✓	✓	✗	✗	✗	✗	✓	✗	✗	✓

ios to simulate the bot activity, and six scenarios to simulate the benign vehicular network traffic. In every scenario, the simulation runs for 500 seconds with 40 nodes moving according to the random way point model with a speed of 20 m/s and no pause time within the Manhattan map (downloaded it from OpenStreetMap [69]). The WiFi is 802.11p, the transmit power is set to 20 dBm, and the default propagation model is Two-Ray Ground.

The first and second scenarios correspond to zero-day DDoS attacks applicable only in vehicular networks environment (highlighted earlier in section 4). In these scenarios, the victim nodes are node 0, node 6, and node 10; while the other nodes are the attackers. As regards the GPS tracking and theft of information scenarios, node 20 represent the bot master, and nodes 7, 8, and 9 are the victims. In the GPS tracking scenario, victim nodes send theirs GPS information to the bot master in real time, while in the information theft scenario, victim nodes send the recorded audio files periodically. For the benign traffic scenarios, we have simulated in each scenario the applications that have common features according to their characteristics, and in which the 40 nodes participate.

We used Network Simulator version 3 (NS3) [70], and the Simulation of Urban MObility (SUMO) package [71]. SUMO is responsible for simulating realistic vehicular traffic while NS3 is used to simulate the communication capabilities of the vehicles with IEEE 802.11p integration. Table 6 summarizes the simulation parameters, and table 7 presents the samples distribution of the network traffic dataset.

6.1.2. Features extraction

After collecting the network traffic generated during simulation (as PCAP files), we have extracted a set of 79 network features (see table 2). To extract features we have used CICFlowMeter [72], a network traffic flow generator distributed by the Canadian Institut for Cyber Security (CIC). It generates bidirectional flows, where the first packet determines the forward (source to destination) and backward (destination to source) directions. Note that TCP flows are usually terminated upon connection teardown (by FIN packet) while UDP flows are terminated by a flow timeout. The flow timeout value can be assigned arbitrarily by the individual scheme e.g., 600 seconds for both TCP and UDP.

Table 6
Simulation parameters

Network Simulator	NS3
Traffic Generator	SUMO
Simulation Area	Manhattan Map
Simulation Time	500 seconds
Number of Nodes	40
Max Speed	20 m/s
MAC/PHY Standard	IEEE802.11p
Traffic Type	WSMP, IP
Bandwidth Channel	CCH, SCH
Propagation Model	Two-ray ground-reflection model
Transmission Power	20 dBm
Packets Size	Depend on the application and protocol
Packets Data Rate	Depend on the application

The list of network flow-based features extracted for each network flow is composed of three categories of features: time, bytes, and packets based features. We believe that time-based features (Flow IAT, Fwd IAT and Idle Time) are useful to detect DoS attacks because time interval between successive packets is too short. Also time-based features allow detection of periodic events such as periodic transfer of collected information in the case of theft of information. The bytes/packets based features allow the detection of large and abnormal traffic increases, which are symptomatic of DoS attacks.

6.1.3. Data pre-processing and features selection

For the data pre-processing step, we did the cleaning and the normalization. To check missing values and deal with them, we used one of the python programming language functions named Dropna(). Dropna removes a row or a column from a data frame, which has a NaN or no values in it. Moreover, to deal with the huge differences between magnitude, units, and range in the generated dataset, we used the feature scaling. The feature scaling aims to put all the values

Table 7
Statistics of normal and attack samples of network traffic dataset

Samples		Nb			%		
		Train	Test	Total	Train	Test	Total
Benign		909	606	1515	31.14%	20.76%	51.90%
Malicious	GPS Tracking	295	197	492	10.11%	6.74%	16.85%
	Phishing Attack	191	128	319	6.55%	4.37%	10.92%
	WSMP-Flood	182	121	303	6.23%	4.15%	10.38%
	Geo-WSMP-Flood	174	116	290	5.96%	3.97%	9.93%
Total		1751	1168	2919	60%	40%	100%

Table 8
List of selected network features

Forward selection features set	LinearSVC features set
Flow Duration, Tot Fwd Pkts, Flow Pkts/s, Flow IAT Mean, Flow IAT Std, Flow IAT Max, Flow IAT Min, Fwd IAT Tot, Fwd IAT Mean, Fwd IAT Std, Fwd IAT Max, Fwd IAT Min, Fwd Pkts/s, Bwd Pkts/s, Down/Up Ratio, Idle Mean, Idle Std, Idle Min.	Tot Fwd Pkts, Tot Bwd Pkts, TotLen Fwd Pkts, Flow Byts/s, Flow Pkts/s, Flow IAT Mean, Flow IAT Std, Flow IAT Max, Flow IAT Min, Fwd IAT Mean, Fwd IAT Std, Fwd IAT Max, Fwd IAT Min, Fwd Header Len, Fwd Pkts/s, Bwd Pkts/s, Pkt Size Avg, Bwd Blk Rate Avg, Init Fwd Win Byts, Init Bwd Win Byts, Active Mean, Active Max.

in the data set between 0 and 1, in order to make the features more consistent with each other and to make the training step less sensitive to this problem.

We have used in this research two types of feature selection algorithms. The first one is forward selection [73], which belongs to the wrappers class. Forward selection is an iterative algorithm that starts with an empty set of feature. In each iteration, it adds the best feature that improves the model until the addition of a new feature does not improve the performance of the model. The Forward feature selection algorithm has reduced the number of features from 37 to 18 features as shown in table 8. The second feature selection algorithm is the Linear Support Vector Classifier LinearSVC [74], which belongs to the embedded features selection algorithms. LinearSVC is an algorithm that gives to each feature a coef_ or feature_importances_ attribute. All the features are considered unimportant at the beginning and it gives them values under a threshold parameter. After that, it uses built-in heuristics for finding the threshold of every feature using a string argument. LinearSVC has reduced the number of features from 37 to 22 as shown in table 8. The best results were achieved when using the subset of features giving by the Forward selection algorithm (as shown in the results section).

6.2. In-vehicule traffic dataset

We have used in our experimentation the dataset built by Song et al. [67]. The authors used Hyundais YF Sonata as a testing vehicle. They connected a Raspberry Pi3 with CAN bus through the OBD-II port (Figure 3), and connect the Raspberry to a laptop computer through WiFi. They generated an in-Vehicle dataset that contains normal CAN frames and other malicious (DoS frames, Fuzzy attack, RPM and Gear). The dataset is available online [75], labelled and in CSV format. We carried the same preprocessing operation of cleaning and normalization described in the previous section. Table 9 presents samples distribution of the in-vehicle traffic dataset.

6.3. Results and discussion

To build the classification models for both analyzers (network and in-vehicule), we apply the supervised machine learning algorithms described in section 5.3 with their default paramaters. For each analyzer we choose the algorithm that gives the best performances. We train, validate and test the two classification models separately (network and in-vehicule analyzer). From each dataset, we take 60% of the dataset to train and validate the classification model through a 10-fold cross-validation, and the remainder 40% for testing the model. Six common metrics, Accuracy, Precision, Recall, F1_score, False Positive rate and false negative rate have been selected to evaluate the classification performances, the aforementioned metrics can be calculated as follows:

$$Accuracy = (TP+TN)/(TP+FP+FN+TN) \quad (7)$$

$$Precision = TP/(TP+FP) \quad (8)$$

$$Recall = TP/(TP+FN) \quad (9)$$

$$\text{F1\_score} = (2*(\text{precision}*\text{recall}))/(\text{precision}+\text{recall}) \quad (10)$$

$$FPR = FP/(TN+FP) \quad (11)$$

Table 9
Statistics of normal and attack samples of in-vehicle traffic dataset

Samples		Nb			%		
		Train	Test	Total	Train	Test	Total
Benign		1239	826	2065	26.93%	17.95%	44.88%
Malicious	DoS Attack	504	336	840	10.95%	7.3%	18.25%
	Fuzzy Attack	304	203	507	6.61%	4.40%	11.01%
	Gear Attack	264	176	440	5.74%	3.82%	9.56%
	RPM Attack	449	300	749	9.76%	6.51%	16.27%
Total		2760	1841	4601	60%	40%	100%

Table 10
Network traffic binary classification using AntibotV

Algorithm	Classe	Precision	Recall	F1-score	FPR	FNR	Accuracy
KNN	Benign traffic	98,5%	99,6%	99,0%	2,3%	0,3%	98,70%
	Malicious traffic	74,53%	72,15%	73,23%	0,05%	27,83%	
	Average Value	86,5%	85,9%	86,1%	1,2%	14,1%	
Neural Netowrks	Benign traffic	97,20%	98,60%	97,90%	4,30%	1,30%	97,30%
	Malicious traffic	77,30%	74,94%	75,94%	1,04%	25,02%	
	Average Value	87,25%	86,77%	86,92%	2,67%	13,16%	
Decision Tree	Benign traffic	99,50%	99,60%	99,50%	0,70%	0,30%	99,40%
	Malicious traffic	82,88%	79,70%	80,88%	0,03%	20,28%	
	Average Value	91,19%	89,65%	90,19%	0,36%	10,29%	
Random Forest	Benign traffic	99,30%	99,50%	99,40%	1,00%	0,40%	99,30%
	Malicious traffic	74,80%	74,70%	74,75%	0,05%	25,28%	
	Average Value	87,05%	87,10%	87,08%	0,53%	12,84%	
SVM	Benign traffic	96,40%	99,40%	97,80%	5,80%	0,50%	97,20%
	Malicious traffic	74,28%	66,75%	69,68%	0,13%	33,23%	
	Average Value	85,34%	83,08%	83,74%	2,96%	16,86%	
Naive Bayes	Benign traffic	96,60%	83,60%	89,60%	4,50%	16,30%	87,70%
	Malicious traffic	65,95%	86,23%	71,48%	2,73%	13,73%	
	Average Value	81,28%	84,91%	80,54%	3,61%	15,01%	

$$FNR = FN/(TP + FN) \tag{12}$$

where TP, FP, TN and FN denote true positive, false positive, true negative, and false negative, respectively.

6.3.1. Malicious network traffic detection

In this experiment, we evaluate the performances of AntibotV on classifying a network connection as legitimate or as a malicious. As we can see from table 10, all the classifiers show high accuracy (> 85%), the highest accuracy is achieved by decision tree (99.4%) followed by random forest (99.3%), while naive bayes presents the lowest accuracy. However, accuracy is not enough to evaluate and select the best classification model. Therefore, other important metrics such as recall (detection rate), precision, false negative rate (FNR) and false positive rate (FPR) need be taken into consideration. Figure 8 compare between the recall, F1-score, FNR, and FPR for benign and malicious traffic.

Decision tree presents the best average recall, however, the malicious detection rate is found to be 79,70%, which represents an intolerable false negative rate (>20 %). Although, the decision tree is able to recognize the legitimate traffic with high precision, the malicious traffic detection performance is not promising. The poor detection rate of malicious traffic is due to heterogeneity of legitimate traffic. Unlike the other types of networks, in vehicular networks there are two types of network traffic: IP and WSMP. The two network traffic show different traffic patterns influenced by several factors such as: context of use, duration, size of packets, etc. This difference represents a challenge on training the classification model, and subsequently mislead the classifier on differentiating between legitimate WSMP and malicious traffic.

To overcome the aforementioned issue, we create two classes of legitimate traffic by separating WSMP and IP traffic. We believe training the classification model with two separated categories of legitimate traffic, will reduce the false negative rate, and thus improve the malicious traffic detection rate. In addition, to provide a suitable response measure, the detection framework needs to identify the type of attacks. Therefore, we separate the malicious traffic into different classes: GPS tracking, WSMP flood, phishing, and Geo flood. From table 11, we can see a significant improvement in malicious traffic detection rate, the recall has increased from 79.7% to 97.59%. Figure 9 compare between

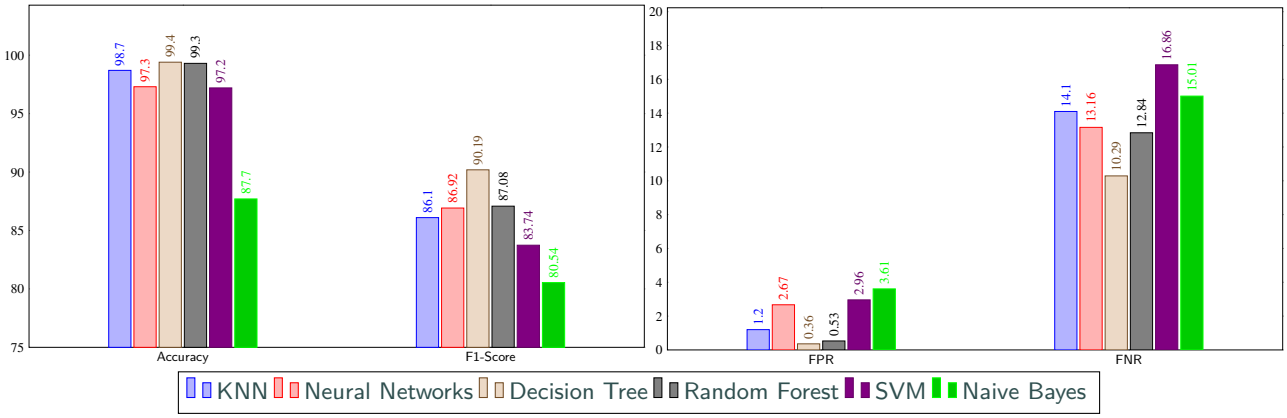

Figure 8: Network traffic binary classification results: Left - Accuracy and F1-score. Right - FPR and FNR

Table 11
Network traffic multiclass results: AntibotV based on Decision Tree classifier

Classes		Precision	Recall	F1-score	FPR	FNR	Accuracy
Benign	WSMP Traffic	99,06%	97,70%	98,38%	0,16%	2,29%	99,40%
	IP Traffic	99,57%	99,71%	99,64%	0,39%	0,28%	
Malicious	GPS Tracking	99,61%	100,00%	99,80%	0,08%	0,00%	
	Phishing Attack	99,47%	100,00%	99,73%	0,07%	0,00%	
	WSMP Flood	97,43%	98,70%	98,06%	0,14%	1,29%	
	GeoFlood	100,00%	91,66%	95,65%	0,00%	8,30%	
Average Value		99,19%	97,96%	98,54%	0,14%	2,03%	

the recall, F1-score, FNR, and FPR for each traffic class separately. We see that decision tree achieves the best performances. Although, the slight decrease, the recall of the benign class remains high (>98). The detection rate of Geo flood attack is not as good as the other attacks. This can be explained by the fact that the majority of security applications on the WSMP protocol use broadcast, which is quite similar to Geo flood attack.

According to the aforementioned results, it is important to note that Geo WSMP flood attack represent the most challenging attack to detect. The Decision Tree achieved the highest detection rate with the lowest false positive rate. Besides detecting malicious traffic, it also delivers high performances on identifying the type of benign (WSMP, IP) and malicious traffic (GPS tracking, phishing, WSMP flood, Geo flood). Therefore, the decision tree is the best classifier to be selected for the network analyzer module.

6.3.2. Malicious in-vehicle traffic detection

We evaluate the performance of AntibotV on detecting and identifying in-vehicle attacks that can be carried out by a bot malware. First, we carry out a binary classification

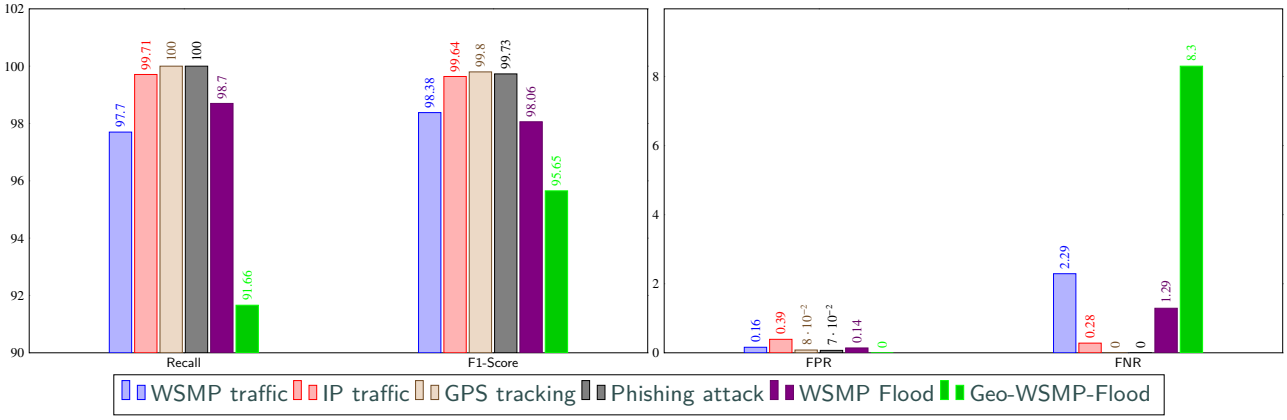

Figure 9: Network Traffic multiclass classification results. Decision Tree Algorithm: Left - Recall, F1-Score. Right - FPR and FNR

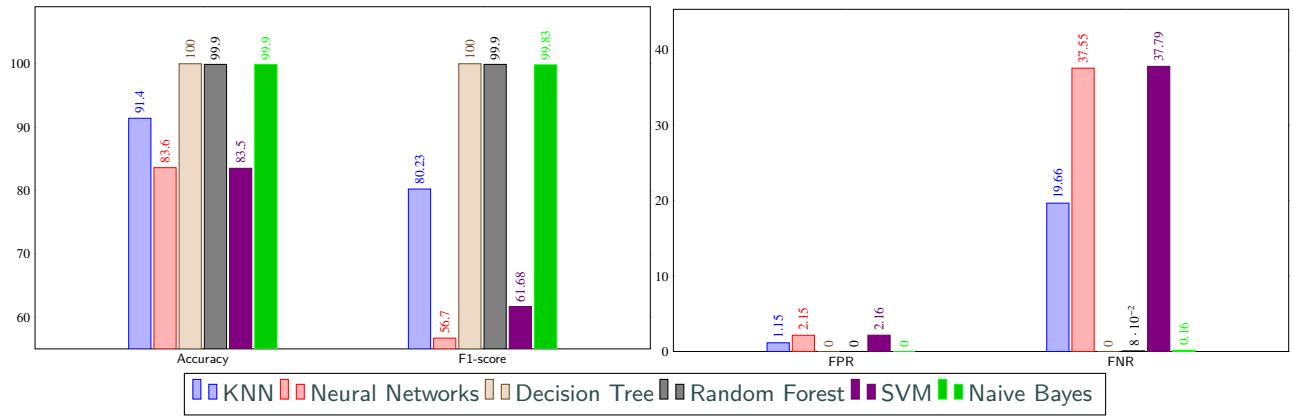

Figure 10: In-Vehicle Binary classification results: Left:Accuracy and F1-score. Right: FPR and FNR

Table 12
In-Vehicle traffic binary classification using AntibotV

Algorithm	Classe	Precision	Recall	F1-score	FPR	FNR	Accuracy
KNN	Benign frames	99,90%	100,00%	99,90%	0,10%	0,00%	91,40%
	Malicious frames	61,30%	60,58%	60,55%	2,20%	39,33%	
	Average value	80,60%	80,29%	80,23%	1,15%	19,66%	
Neural Networks	Benign frames	100,00%	100,00%	100,00%	0,00%	0,00%	83,60%
	Malicious frames	12,48%	24,85%	13,40%	4,30%	75,10%	
	Average value	56,24%	62,43%	56,70%	2,15%	37,55%	
Decision Tree	Benign frames	100,00%	100,00%	100,00%	0,00%	0,00%	100,00%
	Malicious frames	100,00%	100,00%	100,00%	0,00%	0,00%	
	Average value	100,00%	100,00%	100,00%	0,00%	0,00%	
Random Forest	Benign frames	100,00%	100,00%	100,00%	0,00%	0,00%	99,90%
	Malicious frames	99,83%	99,83%	99,80%	0,01%	0,15%	
	Average value	99,91%	99,91%	99,90%	0,00%	0,08%	
SVM	Benign frames	100,00%	100,00%	100,00%	0,00%	0,00%	83,50%
	Malicious frames	23,33%	24,33%	23,35%	4,33%	75,58%	
	Average value	61,66%	62,16%	61,68%	2,16%	37,79%	
Naive Bayes	Benign frames	100,00%	100,00%	100,00%	0,00%	0,00%	99,90%
	Malicious frames	99,68%	99,65%	99,65%	0,00%	0,33%	
	Average value	99,84%	99,83%	99,83%	0,00%	0,16%	

(benign / malicious) using the supervised machine learning algorithms described earlier in section 5.3, then the best classification algorithm is assessed on identifying the type of attack. The accuracy barcharts of the figure 10 clearly show that the proposed framework gives high performances on detecting and identifying the in-vehicle traffic, with over than 90% for the KNN, decision tree, random forest, and naive bayes algorithms. The results of binary classification are presented in table 12. The decision tree is found to deliver the maximum detection rate for both benign and malicious frames. KNN, Neural Networks and SVM algorithms failed to detect the class of malicious traffic, with a poor detection rate (between 25% and 61%). From table 13, we ca see that decision tree can perfectly identify the different types of in-vehicle attacks. These results confirm the efficiency of tree based algorithms for intrusion detection. Therefore, the decision tree is the best classifier to be selected for the in-vehicle analyzer module.

6.4. Discussion and comparison

We compared our solution against the solution presented in [7] that tackled the detection of vehicular botnets, the authors used anomaly detection technique to allow detection of new forms of injected BSMs messages. The detection solution assumes a specific botnet communication protocol (GHOST), which makes it unable to detect botnet using different communication protocol. The proposed framework AntibotV does not suppose a particular botnet communication protocol, so the detection approach works independently of the botnet communication protocol. Limiting detection to monitoring only network communication, makes [7] unable to detect in-vehicle attacks. Thanks to two level monitoring, AntibotV can also detect in-vehicle attacks. Compared to [7], AntibotV delivers better accuracy and lower false positive rate (see table 14). Seo et al. [67] considered in-vehicle threats, they proposed an anomaly detection framework based on convolutional neural network and deep neural network. Although, AntibotV delivers better performances, the proposed approach [67] can detect unseen at-

Table 13
In-Vehicle multiclass results: AntibotV based on Decision Tree classifier

Classes		Precision	Recall	F1-score	FPR	FNR	Accuracy
Benign frames		100,00%	100,00%	100,00%	0,00%	0,00%	100,00%
Malicious frames	DoS Attack	100,00%	100,00%	100,00%	0,00%	0,00%	
	Fuzzy Attack	100,00%	100,00%	100,00%	0,00%	0,00%	
	Gear Attack	100,00%	100,00%	100,00%	0,00%	0,00%	
	RPM Attack	100,00%	100,00%	100,00%	0,00%	0,00%	
Average Value		100,00%	100,00%	100,00%	0,00%	0,00%	

Table 14
Comparison among vehicular botnets detection systems

	AntibotV (our work)	[7]	[67]	[22]
Threat level	Network & In-vehicle	Network	In-vehicle	Network
Machine learning technique	Classification	Anomaly detection	Anomaly detection	Time-series analysis
Algorithm	Decision Tree	3D DBSCAN	CNN & DNN	XGBoost
Dataset	Our own network traffic dataset, and [67] in-vehicle traffic dataset	Their own dataset	Their own dataset	Their own dataset
Ressources requirement	Low	Low	High	Low
Accuracy	99.40% & 100%	77%	97.53%	97.6%
Detection rate (Recall)	97.96% & 100%	NA	98.65%	NA
Precision	99.19% & 100%	NA	97.63%	NA
FPR	0.14% & 0%	NA	NA	3.29%

tacks, but to the detriment of false alerts. Furthermore, deep learning techniques requires large amount of computational and memory resources, which could be constraining in vehicular context. The authors in [22] have proposed a mechanism which, detects, mitigates and traces back the location of a Low-Rate DDoS attacker. If the DDoS attack is caused by a botnet, the traceability mechanism can be used to identify bot nodes in the network. However, if the botnet is used to apply other threats (e.g. information theft), the proposed detection techniques will not be effective anymore, unlike our proposed framework (Antibotv), which is able to deal different types of botnet threats at different levels. Table 14 provides a qualitative comparison between AntibotV and [7, 22] (different databases), and a quantitative comparison with [67] (the same in-vehicular traffic database).

7. Conclusion

In this paper, we have proposed AntibotV, a multilevel behaviour-based framework to detect vehicular botnets. We have considered new zero day attacks, as well as a wide range of DoS and in-vehicle attacks. The proposed framework monitors the vehicles activity at network and in-vehicle levels. To build the detection system, we have collected network traffic data of legitimate and malicious applications. Then, training using decision tree a new classifier with a set of features that we have extracted and selected. Likewise, we have trained a decision tree with in-vehicle data. The experimental results showed that AntibotV outperforms existing solutions, it achieves a detection rate higher than 97% and a false positive rate lower than 0.14%. To add a realistic dimension to our framework, the ideal was to do the training and validation with a dataset generated from a realistic testbed. In our futur work, we will work on the generation of a realistic dataset, which takes into account different factors, such as the impact of the environment (urban, rural and highway), plus the implementation of different malicious behavior of a bot vehicle. In a second step, we can use AntibotV generated from real data to design a simple system to report feedback from the driver, instead of using only raw features directly. Moreover, AntibotV has a fundamental limitation in detecting unlearned types of attacks because it is based on supervised learning. To solve this problem, additional research on unknown attack detection is required using advanced learning techniques.

8. Acknowledgment

This research is a result from PRFU project C00L07UN23 0120180009 funded in Algeria by La Direction Générale de la Recherche Scientifique et du Développement Technologique (DGRSDT).

References

[1] Eun-Kyu Lee, Mario Gerla, Giovanni Pau, Uichin Lee, and Jae-Han Lim. Internet of vehicles: From intelligent grid to autonomous cars and vehicular fogs. *International Journal of Distributed Sensor Networks*, 12(9):1550147716665500, 2016.

[2] Hariharan Krishnan, Fan Bai, and Gavin Holland. Commercial and public use applications. In *Vehicular Networking*, pages 1–28. John Wiley & Sons, Ltd, April 2010.

[3] Martina Stoyanova Todorova and Stamelina Tomova Todorova. Ddos attack detection in sdn-based vanet architectures. *no. June*, page 175, 2016.

[4] Jun Zhou, Zhenfu Cao, Xiaolei Dong, and Athanasios V Vasilakos. Security and privacy for cloud-based iot: Challenges. *IEEE Communications Magazine*, 55(1):26–33, 2017.

[5] Parul Tyagi and Deepak Dembla. Investigating the security threats in vehicular ad hoc networks (vanets): towards security engineering

for safer on-road transportation. In *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2084–2090. IEEE, 2014.

[6] Jiajia Liu, Shubin Zhang, Wen Sun, and Yongpeng Shi. In-vehicle network attacks and countermeasures: Challenges and future directions. *IEEE Network*, 31(5):50–58, 2017.

[7] Mevlut Turker Garip, Peter Reiher, and Mario Gerla. Ghost: Concealing vehicular botnet communication in the vanet control channel. In *2016 International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 1–6. IEEE, 2016.

[8] Mevlut Turker Garip, Jonathan Lin, Peter Reiher, and Mario Gerla. Shieldnet: An adaptive detection mechanism against vehicular botnets in vanets. In *2019 IEEE Vehicular Networking Conference (VNC)*, pages 1–7. IEEE, 2019.

[9] Ming-Chiao Chen and Teng-Wen Chang. Introduction of vehicular network architectures. In *Telematics Communication Technologies and Vehicular Networks: Wireless Architectures and Applications*, pages 1–14. IGI Global, 2010.

[10] Jun Zhou, Xiaolei Dong, Zhenfu Cao, and Athanasios V Vasilakos. Secure and privacy preserving protocol for cloud-based vehicular dtns. *IEEE Transactions on Information Forensics and Security*, 10(6):1299–1314, 2015.

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

[12] Shereen AM Ahmed, Sharifah HS Ariffin, and Norsheila Fisal. Overview of wireless access in vehicular environment (wave) protocols and standards. *environment*, 7:8, 2013.

[13] 1609.4-2016 - ieee standard for wireless access in vehicular environments (wave) – multi-channel operation.

[14] Caixia Song. Performance analysis of the ieee 802.11 p multichannel mac protocol in vehicular ad hoc networks. *Sensors*, 17(12):2890, 2017.

[15] 1609.2-2016 - ieee standard for wireless access in vehicular environments–security services for applications and management messages.

[16] Chris Hedges and Frank Perry. Overview and use of sae j2735 message sets for commercial vehicles. Technical report, SAE Technical Paper, 2008.

[17] Automotive electronic control unit market size, share, trends analysis report by application, by propulsion type, by capacity, by vehicle type, by region, and segment forecasts, 2019 - 2025. https://www.grandviewresearch.com/industry-analysis/automotive-ecu-market. Accessed: 2021-05-05.

[18] Controller area network. http://www.esd-electronics-usa.com/Controller-Area-Network-CAN-Introduction.html. Accessed: 2021-01-09.

[19] Botnet mirai. https://www.cloudflare.com/learning/ddos/glossary/mirai-botnet/. Accessed: 2021-01-08.

[20] 9 of history's notable botnet attacks. https://www.whiteops.com/blog/9-of-the-most-notable-botnets. Accessed: 2021-01-09.

[21] João Henrique Corrêa, Patrick M Ciarelli, Moises RN Ribeiro, and Rodolfo S Villaça. Ml-based ddos detection and identification using native cloud telemetry macroscopic monitoring. *Journal of Network and Systems Management*, 29(2):1–28, 2021.

[22] Neha Agrawal and Shashikala Tapaswi. An sdn-assisted defense mechanism for the shrew ddos attack in a cloud computing environment. *Journal of Network and Systems Management*, 29(2):1–28, 2021.

[23] Mohammad Alhisnawi and Mahmood Ahmadi. Detecting and mitigating ddos attack in named data networking. *Journal of Network and Systems Management*, 28:1343–1365, 2020.

[24] Rabah Rahal, Abdelaziz Amara Korba, and Nacira Ghoualmi-Zine. Towards the development of realistic dos dataset for intelligent transportation systems. *Wireless Personal Communications*, 115(2):1415–1444, 2020.

[25] Yazan Otoum and Amiya Nayak. As-ids: Anomaly and signature based ids for the internet of things. *Journal of Network and Systems Management*, 29(3):1–26, 2021.

[26] Ayodeji Oseni, Nour Moustafa, Helge Janicke, Peng Liu, Zahir Tari, and Athanasios Vasilakos. Security and privacy for artificial intelligence: Opportunities and challenges. *arXiv preprint arXiv:2102.04661*, 2021.

[27] Ximeng Liu, Lehui Xie, Yaopeng Wang, Jian Zou, Jinbo Xiong, Zuobin Ying, and Athanasios V Vasilakos. Privacy and security issues in deep learning: A survey. *IEEE Access*, 2020.

[28] M Dibaei, X Zheng, Y Xia, X Xu, A Jolfaei, A Kashif Bashir, U Tariq, D Yu, and AV Vasilakos. Investigating the prospect of leveraging blockchain and machine learning to secure vehicular networks: A survey. *IEEE: Piscataway, NJ, USA*, 2020.

[29] Rajib Biswas and Sambuddha Roy. Botnet traffic identification using neural networks. *Multimedia Tools and Applications*, pages 1–25, 2021.

[30] Kapil Sinha, Arun Viswanathan, and Julian Bunn. Tracking temporal evolution of network activity for botnet detection. *arXiv preprint arXiv:1908.03443*, 2019.

[31] David Zhao, Issa Traore, Bassam Sayed, Wei Lu, Sherif Saad, Ali Ghorbani, and Dan Garant. Botnet detection based on traffic behavior analysis and flow intervals. *Computers & Security*, 39:2–16, 2013.

[32] W Timothy Strayer, David Lapsely, Robert Walsh, and Carl Livadas. Botnet detection based on network behavior. In *Botnet detection*, pages 1–24. Springer, 2008.

[33] Supranamaya Ranjan. Machine learning based botnet detection using real-time extracted traffic features, March 25 2014. US Patent 8,682,812.

[34] Supranamaya Ranjan and Feilong Chen. Machine learning based botnet detection with dynamic adaptation, March 19 2013. US Patent 8,402,543.

[35] Nicoló Andronio, Stefano Zanero, and Federico Maggi. Heldroid: Dissecting and detecting mobile ransomware. In *International Symposium on Recent Advances in Intrusion Detection*, pages 382–404. Springer, 2015.

[36] William Enck, Peter Gilbert, Seungyeop Han, Vasant Tendulkar, Byung-Gon Chun, Landon P Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N Sheth. Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones. *ACM Transactions on Computer Systems (TOCS)*, 32(2):1–29, 2014.

[37] Jianbing Ni, Kuan Zhang, and Athanasios V Vasilakos. Security and privacy for mobile edge caching: Challenges and solutions. *IEEE Wireless Communications*, 2020.

[38] Machigar Ongtang, Stephen McLaughlin, William Enck, and Patrick McDaniel. Semantically rich application-centric security in android. *Security and Communication Networks*, 5(6):658–673, 2012.

[39] Min Zhao, Tao Zhang, Fangbin Ge, and Zhijian Yuan. Robotdroid: a lightweight malware detection framework on smartphones. *Journal of Networks*, 7(4):715, 2012.

[40] Sven Nomm and Hayretdin Bahsi. Unsupervised anomaly based botnet detection in IoT networks. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, December 2018.

[41] Raihana Syahirah Binti Abdullah, MA Faizal, and Zul Azri Muhamad Noh. Multivariate statistical analysis on anomaly p2p botnets detection. *EVOLUTION*, 8(12), 2017.

[42] Basudeb Bera, Sourav Saha, Ashok Kumar Das, and Athanasios V Vasilakos. Designing blockchain-based access control protocol in iot-enabled smart-grid system. *IEEE Internet of Things Journal*, 8(7):5744–5761, 2021.

[43] Philokypros Ioulianou, Vasileios Vasilakis, Ioannis Moscholios, and Michael Logothetis. A signature-based intrusion detection system for the internet of things. *Information and Communication Technology Form*, 2018.

[44] Owen Dwyer, Angelos Marnerides, Vasileios Giotsas, and Troy Mursh. Profiling iot-based botnet traffic using dns. 2019.

[45] Wanting Li, Jian Jin, and Jong-Hyouk Lee. Analysis of botnet domain names for iot cybersecurity. *IEEE Access*, 7:94658–94665, 2019.

[46] Aaron Ridley, Robert Abbas, and Ponnappan Ponnurangam. Machine

leaning dns data analysis for automated maliciousdomain classification. 2019.
[47] Mohammad Wazid, Ashok Kumar Das, Vivekananda Bhat, and Athanasios V Vasilakos. Lam-ciot: Lightweight authentication mechanism in cloud-based iot environment. *Journal of Network and Computer Applications*, 150:102496, 2020.
[48] Srinivas Jangirala, Ashok Kumar Das, Mohammad Wazid, and Athanasios V Vasilakos. Designing secure user authentication protocol for big data collection in iot-based intelligent transportation system. *IEEE Internet of Things Journal*, 2020.
[49] Mevlut Turker Garip, Peter Reiher, and Mario Gerla. Botveillance: A vehicular botnet surveillance attack against pseudonymous systems in vanets. In *2018 11th IFIP Wireless and Mobile Networking Conference (WMNC)*, pages 1–8. IEEE, 2018.
[50] Mevlut Turker Garip, Mehmet Emre Gursoy, Peter Reiher, and Mario Gerla. Congestion attacks to autonomous cars using vehicular botnets. In *NDSS Workshop on Security of Emerging Networking Technologies (SENT), San Diego, CA*, 2015.
[51] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, Tadayoshi Kohno, et al. Comprehensive experimental analyses of automotive attack surfaces. In *USENIX Security Symposium*, volume 4, pages 447–462. San Francisco, 2011.
[52] Siri. https://www.apple.com/siri/. Accessed: 2021-01-08.
[53] Ben Lovejoy, Ben Lovejoy, Ben Lovejoy, and Eu. Malicious siri commands can be hidden in music and innocuous-sounding speech recordings, May 2018.
[54] Takeshi Sugawara, Benjamin Cyr, Sara Rampazzi, Daniel Genkin, and Kevin Fu. Light commands: Laser-based audio injection attacks on voice-controllable systems. 2019.
[55] Margi Murphy. How google is secretly recording you through your mobile, monitoring millions of conversations every day and storing the creepy audio files, Aug 2017.
[56] Kévin Thomas, Hacéne Fouchal, Stéphane Cormier, and Francis Rousseaux. C-its communications based on ble messages. In *GLOBECOM 2020-2020 IEEE Global Communications Conference*, pages 1–7. IEEE, 2020.
[57] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, et al. Experimental security analysis of a modern automobile. In *IEEE Symposium on security and privacy*, volume 10. Oakland, 2010.
[58] Mark Wood and Michael Erlinger. Intrusion detection message exchange requirements. *IETF, draft-ietf-idwg-requirements-10*, 2002.
[59] Ipv6 flow label specification. https://tools.ietf.org/html/rfc3697/. Accessed: 2021-01-08.
[60] Controller area network (can) link laye. https://erg.abdn.ac.uk/users/gorry/eg3576/CAN-link.html. Accessed: 2021-01-08.
[61] Sukru Yaren Gelbal, Sheng Zhu, Gokul Arvind Anantharaman, Bilin Aksun Guvenc, and Levent Guvenc. Cooperative collision avoidance in a connected vehicle environment. Technical report, SAE Technical Paper, 2019.
[62] Vinh-Thong Ta and Amit Dvir. A secure road traffic congestion detection and notification concept based on v2i communications. *Vehicular Communications*, 25:100283, 2020.
[63] Ying-ji Liu, Yu Yao, Cheng-xu Liu, Lin-tao Chu, and Xu Liu. A remote on-line diagnostic system for vehicles by integrating obd, gps and 3g techniques. In *Practical Applications of Intelligent Systems*, pages 607–614. Springer, 2011.
[64] Songbo Tan. Neighbor-weighted k-nearest neighbor for unbalanced text corpus. *Expert Systems with Applications*, 28(4):667–671, 2005.
[65] Henk Pelk. Machine learning, neural networks and algorithms, Sep 2017.
[66] S Abirami and P Chitra. Energy-efficient edge based real-time healthcare support system. In *Advances in Computers*, volume 117, pages 339–368. Elsevier, 2020.
[67] Eunbi Seo, Hyun Min Song, and Huy Kang Kim. Gids: Gan based intrusion detection system for in-vehicle network. In *2018 16th Annual Conference on Privacy, Security and Trust (PST)*, pages 1–6. IEEE, 2018.
[68] Hariharan Krishnan, Fan Bai, and Gavin Holland. Commercial and public use applications. *Vehicular Networking*, pages 1–28, 2010.
[69] Openstreetmap. https://www.openstreetmap.org/. Accessed: 2021-01-08.
[70] Network simulator 3. https://www.nsnam.org/. Accessed: 2021-01-08.
[71] Simulation of urban mobility. http://sumo.sourceforge.net/. Accessed: 2021-01-08.
[72] Cicflowmeter. http://netflowmeter.ca/. Accessed: 2019-07-08.
[73] Forward selection algorithm. http://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/. Accessed: 2019-07-08.
[74] Linear-svc. https://scikit-learn.org/stable/modules/feature_selection.html}l1-based-feature-selection. Accessed: 2019-07-08.
[75] Car-hacking dataset. http://ocslab.hksecurity.net/Datasets/CAN-intrusion-dataset. Accessed: 2021-01-08.

Rabah RAHAL is a Doctoral student in Badji Mokhtar University Annaba Algeria. He is a member of the Laboratory of Computer Networks and Systems. His research interests include vehicular networks, network security, intrusion and malwares detection, machine learning, anomaly detection, cryptography, security analysis. He does research in cyberattacks detection in vehicular networks, and the generation of realistic DoS dataset for intelligent transportation systems.

Abdelaziz Amara korba received his Ph.D. degree from Badji Mokhtar Annaba University, Algeria, in 2016. Currently, he is an Associate Professor in the department of computer science at the University of Badji Mokhtar Annaba, and he is member of Networks and Systems Laboratory (LRS). Dr. Amara korba has made contributions in the fields of network security, intrusion and cyberattacks detection. His research interests include cybersecurity, intrusion detection, anomaly detection, IoT, botnets. He served as TPC in many international conferences worldwide (EUSPN, AINIS, ICCE,..), he serves as an associate editor for the International Journal of Smart Security Technologies.

Nacira GhoualmiZine is a Professor in Computer Sciences and has been a Lecturer in the Department of Computer Science at Badji Mokhtar University, Annaba, Algeria since 1985. She is the Head of the Master and Doctoral option entitled Network and Computer Security, and Head of a Laboratory of Computer Networks and Systems. Her research includes cryptography, computer security, intrusion detection system, wireless networks, distributed multimedia applications, quality of service, security in the protocol, and optimisation in networks.

Yacine CHALLAL received the PhD degree in computer science from Compigne University of Technology (UTC), France, in 2005. Since September 2007, he has been an Assistant Professor at UTC. He is currently Professor Ecole nationale Supérieure dInformatique, Algiers, Algeria. His research interests include network security, wireless communication security, cyber-physical systems security, wireless sensor networks, IoT, Cloud computing and fault tolerance in distributed systems. He served as managing guest editor of Ad Hoc Networks for a special issue on Internet of Things security and privacy, and served as guest editor of Wiley Security and Communication Networks journal on Security, Privacy and Trust in Emerging Wireless Networks.

Yacine Ghamri-Doudane is currently Full Professor at the University of La Rochelle (ULR) in France, and the director of its Laboratory of Informatics, Image and Interaction (L3i). Before that, Yacine held an Assistant/Associate Professor position at ENSIIE, a major French post-graduate school located in Evry, France, and was a member of the Gaspard-Monge Computer Science Laboratory (LIGM UMR 8049) at Marne-la-Vallée, France. From February 2011 till July 2012, he was regularly visiting the Performance Engineering Laboratory of University College Dublin, Dublin, Ireland. Yacine received an engineering degree in computer science from the National Institute of Computer Science (INI), Algiers, Algeria, in 1998, an M.S. degree in signal, image and speech processing from the National Institute of Applied Sciences (INSA), Lyon, France, in 1999, a Ph.D. degree in computer networks from University Pierre Marie Curie, Paris 6, France, in 2003, and a Habilitation to Supervise Research (HDR) in Computer Science from Université Paris-Est, in 2010. His current research interests lays in the area of wireless networking and mobile computing with a current emphasis on topics related to the Internet of Things (IoT), Wireless Sensor Networks (WSN), Vehicular Networks as well as Digital Trust. Yacine holds three (3) international patents and he authored or co-authored seven (7) book chapters, more than 30 peer-reviewed international journal articles and more than 130 peer-reviewed conference papers. Since 1999, he participated or still participates to several national and European-wide research projects in his area of interests. Among them two regional research projects (one ongoing), four national-wide research projects (one ongoing), ten European-wide research projects (two on-going) as well as three EU COST Actions (one ongoing). As part of his professional activities linked to the computer networking research community, Yacine also acted as the Chair the IEEE Communications Society (ComSoc) Technical Committee on Information Infrastructure Networking (TCIIN previously TCII) from January 2010 till December 2013 and also chaired the IEEE ComSoc Humanitarian Communications Technologies Ad hoc Committee (HCTC) from January 2012 till December 2015. He is or had been an editorial board member of the Elsevier JNCA, Elsevier ComNet, Springer AoT Journals, Wiley WCMC, Guest Editor of IEEE ComMag, IEEE IoT Journal, Springer/EURASIP WCN Journal, co-Editor in Chief of the Elsevier/KICS ICT Express Journal as well as the founding editor-in-chief of the IEEE ComSoc Ad Hoc and Sensor Network Technical Committee (AHSN TC) Newsletter. Among other conference involvements, he acted or is still currently acting as the TPC Chair of IEEE CCNC 2015, Symposium co-Chair in IEEE ICC 2009, 2010, 2012 and 2018 as well as IEEE GLOBECOM 2012 and 2015. He is a Member of IEEE.