OBD_SecureAlert: An Anomaly Detection System for Vehicles

Sandeep Nair Narayanan, Sudip Mittal & Anupam Joshi
{sand7, smittal1, joshi}@umbc.edu
University of Maryland, Baltimore County, Baltimore, MD 21250, USA

***Abstract*—Vehicles can be considered as a specialized form of Cyber Physical Systems with sensors, ECU's and actuators working together to produce a coherent behavior. With the advent of external connectivity, a larger attack surface has opened up which not only affects the passengers inside vehicles, but also people around them. One of the main causes of this increased attack surface is because of the advanced systems built on top of old and less secure common bus frameworks which lacks basic authentication mechanisms. To make such systems more secure, we approach this issue as a data analytic problem that can detect anomalous states. To accomplish that we collected data flowing between different components from real vehicles and using a Hidden Markov Model, we detect malicious behaviors and issue alerts, while a vehicle is in operation. Our evaluations using single parameter and two parameters together provide enough evidence that such techniques could be successfully used to detect anomalies in vehicles. Moreover our method could be used in new vehicles as well as older ones.**

I. Introduction

According to US department of transportation [1], 88% of all people drive and there are 1.9 vehicles on an average in each American household [2]. Vehicles are an integral part of our life and automobile technology has evolved over the past century to address our growing needs. Earlier, a driver had to manually control various functions in a vehicle, but now a lot of these tasks have been delegated to various micro-controllers and electronic chips attached to the vehicle. Modern vehicles are a collection of various Electronic Control Units (ECU), Sensor and Actuators. Some general purpose control units present in a modern vehicle are Anti-lock Brake System (ABS), Adaptive Cruise control, Active Suspension, Active Vibration Control, Entertainment System, Lane Keeping Assist, Electronic Power Steering, Adaptive Front lighting, etc. These ECU's get input from different sensors and perform various mechanical actions using actuators. Apart from doing their own functions, ECU's must communicate between each other so as to efficiently perform their functions. To ease inter-controller communication, a common internal communication bus was introduced. Bosch proposed the *'controller area network (CAN bus)'* and the updated specification, CAN 2.0, was published in the year 1991 [3]. In 1993, the International Organization for Standardization released the CAN standard, ISO 11898 [4], based on CAN 2.0 and was adopted as a standard for inter ECU communication. Since the bus was designed to be fast and simple, it lacked various authentication schemes and non-repudiation mechanisms.

In the *Internet-of-Things* age, when we are increasingly connecting various appliances to the global World Wide Web, cars have not been left behind. Features like remote ignition start, Internet enabled music devices, etc. are already present in new vehicle models sold by various vendors. These features are important advertising points used by car manufacturers to set their vehicles apart. Adding these 'ease of use' features can open multitude of attack surfaces which can be used to directly affect the functionality of critical ECUs. Various 'after the market' and plug-n-play devices can make existing on-road vehicles also vulnerable. Hence we will need to address the problem of securing these IoT enabled cars as new Internet enabled functionality will be added in the near future.

CAN bus is a broadcast bus, where each connected ECU pushes broadcast messages on it. These broadcast CAN messages don't have explicit information about which ECU generated the message and any message available on the network will considered as 'trusted' by default. As a result if any malicious message is introduced into the network, either by a malicious ECU or an attacker, will also be considered as valid and can result in abnormal behavior. A newer protocol can secure technologies in future cars, but they may not be useful for on-road vehicles. We believe that detecting an attack is the first step to secure them. Hence, we envision an anomaly detection mechanism, OBD-SecureAlert which can detect abnormal activities in new and 'on-road' vehicles. In this paper, we first collected CAN message data from different vehicles and used Hidden Markov Models to generate a model. Our technique will then monitor various CAN messages to detect anomalous states and generate alerts as required. It is to be noted that we are creating an alert system and not a preventive tool. Our system will not issue / process a preventive action. Finally we used a progressive approach for evaluation. We developed model using a single parameter and using 2 parameters together and evaluated them against multiple hand-crafted anomalous scenarios. Our evaluations provide evidence that such a technique can be used to detect attacks on vehicles.

In Section II, we discuss different approaches and techniques trying to address the security issue. Section III, gives a brief description and overview of our system. We present their details in sections IV, V and VI. After discussing our evaluation techniques in section VII, we conclude the paper and discuss future work in section VIII.

II. Related Work

In order to secure any given system there are two methods, one is to prevent the attack and second is to detect and mitigate potential risks. In this section we first discuss various techniques to attack a vehicle. Then we discuss various techniques to secure them.

A. Attack on Vehicles

One of the main reasons which enables attackers to inject potentially malicious messages on CAN bus is that the protocol lacks authentication mechanisms [5]. Researchers tried to address this problem by using cryptography. Wolf et al. [6] looked at the requirements of cryptographic functions for car security. They proposed embedded solutions to add cryptographic functions to different ECU's which can provide security against malicious manipulations. Hoppe et al. [7] describe different attacks which are possible by malicious modifications of ECU code. Using one such modification they were able to open the car windows [8] automatically when it reaches the speed of 200 kilometer per hour. These vulnerabilities are severe since they directly affect passenger safety. In another instance, they hacked comfort control unit so as to control warning lights. Since comfort control ECU is responsible for anti-theft functionality, they could have easily disabled the alarm system to ease unauthorized entry to a vehicle. Researchers were also able to manipulate 'Airbag Control Systems' [7] in vehicles using a similar technique. Koscher et al. [5] experimentally evaluated and demonstrated various problems with the underlying system structure. They illustrated how easy it was for a determined attacker to infiltrate various ECU's and circumvent different security systems. They were able to attack various components like speedometer, lights, brakes and doors locks.

Recently researchers were able to exploit the weakness in a car system build on top of basic CAN network by injecting malicious data into the internal bus. Effects of their hack ranged from acts like switching the lights on, to potentially fatal acts like applying brakes. Charlie Miller and Chris Valasek demonstrated a hack by attaching an external device [9]. They demonstrated their recent work at Blackhat 2015, in which they hacked a Jeep Cherokee [10] remotely without even attaching any external device. More reports [11] have come out listing various vehicles from popular car makers like Volkswagen, Skoda, Volvo, etc. that are vulnerable to another kind of crypto attack on key less entry.

B. Attack Prevention

Hazem and Fahmy [12] proposed LCAP, a light weight CAN authentication protocol to secure CAN bus in which they reduced communication overhead and computational complexity associated with cryptography. In this protocol, they used a 'one-way hash function' to generate a 'magic number' which is selected by the sender and can only be verified by the receiver. Magic number would be sent either using the 'extended identifier field' or as a payload over CAN bus. Another authentication protocol is CANAuth by Herrewege et al. [13], which is backward compatible with the currently used protocol. Apart from proposing a new protocol, they identified different restrictions on CAN bus system like hard real-time constraints, message length restriction, lack of bi-directional communication, etc. In this protocol, authentication data is transmitted out of band which provides a maximum length of 15 bytes for authentication message. LiBrA-CAN by Groza et al. [14] is yet another protocol to secure the CAN bus. Instead of providing independent authentication for each sensor or ECU, they assigned keys for a group of these devices. They employed 'key splitting' and 'MAC mixing' in this protocol to provide security. Apart from basic authentication scheme, they also discussed several variations of their protocol which are broadly classified as 'master oriented authentication schemes' and 'distributed authentication schemes'.

C. Attack Detection and Mitigation

Koscher et al. [15] discusses the importance of 'detection mechanisms' versus 'prevention mechanisms'. They solidify our hypothesis that, operational and economic realities in the domain demands a detection strategy unlike prevention strategies using cryptography mentioned above. Ruta et al. [16] collected OBD data from CAN bus and analyzed it to infer potential risk factors to provide users with warnings. In their method they fused the OBD speed and RPM information with external data like weather information, location information, etc. using simple 'data fusion' algorithms to perform logic based matchmaking. They inferred road and traffic conditions, driving behavior, etc. and generated suggestions to minimize risk factors. For example, their system suggests the driver to use ABS and fog lamps along with slow driving if it detects a foggy weather at a particular location.

D. Hidden Markov Models and their Applications

In our project we use Hidden Markov Models which are quite popular for analyzing time series data. Hidden markov models have been applied to many problems in various fields like finance, bioinformatics, etc. Ziv et al. [17] successfully applied it to analyze time series gene expression data to study a wide range of biological systems. They stress that hidden markov models help them to infer causality from temporal response patterns and address the challenge of handling different non-uniform sampling rates. Zhang et al. [18] used hidden markov models to analyze financial data. They take historical multidimensional and complex nonlinear data from various financial indexes and develop a hidden markov prediction system to find a possible future value of a stock price. Guo et al. [19] used accelerometer and GPS data to develop a movement and behavior model for cattle by using hidden markov models. The authors collected real data for individual cows in the herd and then predicted their movements using machine learning models.

As mentioned above Koscher et al. [15] described the importance of a detection scheme due to practical issues. Moreover the data analytics on OBD data [16] proves useful. This leads us to believe that a machine learning approach can

provide us a method to detect abnormal vehicle behavior using data from CAN bus.

III. System Architecture

Newer protocols require significant modifications to ECU and sensor architecture. Accommodating these changes for future vehicles could cause significant increase in manufacturing cost while for existing cars, modifying their existing components would be harder or economically impractical. Since it is possible to add third party gadgets, even to older cars, we believe that it is extremely important to make the in-vehicular network safe by detecting and possibly mitigating potential attacks. Hence we envision a mechanism which is applicable to older and newer vehicles at the same time with minimum modification to the existing architecture.

In the current architecture different ECU's communicate with each other over CAN bus by broadcasting messages. This communication generates a regular stream of messages on the common bus. For our experiments we analyze message streams from different ECU's: Engine Control Module, Electronic Brake control, Transmission Control, Body Control, Telematics, Radio, etc. We then formulate these sequence of events as a time series machine learning problem and use a model to predict if the vehicle's state is normal or abnormal.

We divide our work into 3 phases which we describe in the following sections:

- Data Collection Phase: This is the first step in which the stream of CAN bus data is collected from real vehicles for analysis. We can employed the OBD-II port present in most vehicles for this purpose.
- Model Generation Phase: In this phase we analyze the collected data and generate a model. Since Hidden Markov Models (HMM) can abstract the time series data, we use them to model this scenario.
- Anomaly Detection Phase: This is the final phase in which we detect anomalous behaviors using generated model and posterior probabilities.

A. System Integration

Our model can be integrated with all current and future car systems as a plug-n-play device or as a system module programmed on the on-board car computer. We can add our system to old cars using their pre-existing OBD port and attaching a small raspberry-pie chip to the OBD port to collect, analyze, and issue alerts. New cars can have this feature pre-installed.

IV. Data Collection

To detect unsafe states of a vehicle, we first collect data from the common bus. When a device want to communicate with other components, it broadcasts a message on to the bus with a specific message ID. The Figure 1 shows intercepted CAN messages. Each message has a specific Message ID (part before semicolon) and message data (part inside square brackets). In a message, each device is identified by a Message ID. It should be noted that each device connected to the CAN bus will receive all messages. At the receiver end, only those devices which need that message process it while others just ignore it. To avoid two devices broadcasting simultaneously, different priority mechanisms are used.

```
01 10: [7E8 04 41 10 01 1E AA AA AA ]
01 44: [7E8 04 41 44 80 00 AA AA AA ]
01 04: [7E8 03 41 04 3F AA AA AA AA ]
01 06: [7E8 03 41 06 84 AA AA AA AA ]
01 07: [7E8 03 41 07 82 AA AA AA AA ]
01 0B: [7E8 03 41 0B 34 AA AA AA AA ]
01 0C: [7E8 04 41 0C 0C 62 AA AA AA ]
```

Fig. 1: Intercepted CAN Messages.

The OBD port is connected to the common bus in order to collect diagnostic information. So we can attach a device to it and extract data for analysis. There are multiple tools in the market like OBDLink Mx, Blue driver, CAN-BUS Shield and ELM 327 clone devices which can be used to collect messages from CAN bus. For our data collection, we used a STN1100 based OBDLink MX. STN1100 is a multi-protocol OBD to UART interpreter integrated circuit. It has a 16 bit processor with inbuilt flash memory and RAM. It supports the complete AT command set (Command set for ELM 327 based chip-set) along with a new set of ST commands. It supports different protocols like ISO 15765-4 (CAN), ISO 11898 (raw CAN) and SAE J1939 (Heavy vehicles). We used the device to interface with OBD port and collected the data on CAN bus.

Using the above mentioned setup, we collected data from vehicle of different manufacturers like "Honda", "Toyota" and "Chevrolet". We faced some practical limitations while collecting data from different cars. Many of these vehicle manufacturers have different mechanisms which hinder direct data collection. Some of the techniques include using multiple CAN buses which are guarded by different gateways. Another challenge is the use of non-standard CAN message ID's by different manufacturers. But these simple techniques won't hinder a malicious attacker. We were able to collect the information from sensors which include speed, load, engine coolant temperature, Engine RPM, Intake air temperature, Absolute throttle position and O2 voltage.

V. Model Generation

We collected data by operating different vehicles and stored it in a text file. Next step in our approach is to analyze the collected data to develop a model which can identify anomalous states. In this project, we use Hidden Markov Models (HMM) to create a model. The intuition behind using this model is described below. We consider the movement of a vehicle as a sequence of states which are dependent on its previous state, like Markov's processes. For example consider a sequence of activities from T_1 to T_{12} as shown in Figure 2. At T_1 speed is zero and the Door is open. At T_2 the door is closed and it starts moving. The car gathers speed gradually till T_6. But at T_7 there is a sudden jump of 85 miles per hour making the speed to 100 mph. At T_8 the

speed of car is 200 miles per hour and the door is open. We can clearly see that the probability of a state change from T_6 to T_7 and T_7 to T_8 are very low. This shows that we can detect anomalous behaviors using a time series analysis. We used HMM's to create a model since they provide a powerful abstraction to predict time series data. To create a HMM model, we generate two set of probabilities, Transition probabilities and Emission probabilities. Transition probability controls how a new state, let's say "S(t)", is chosen from a current state "S(t-1)". Emission probability is the probability that a specific set of observations will be generated given current hidden state "S(t)". During model generation we try to estimate these probabilities using the collected data set.

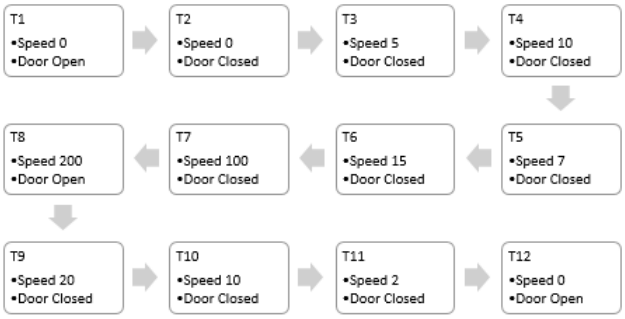

Fig. 2: Sample Car Event Time-line

The first challenge for model generation is how to convert collected data into a series of observations. Our dataset has messages from multiple ECU's. Instead of training the model with absolute values, we chose to use gradients for each observation, since it is the change in observations which alters the state of a vehicle. For example in case of speed, instead of using actual speed, we find speed gradients and train our system for it. The next challenge is on how to accommodate multiple observations as a single vector. We have different types of sensors in a vehicular system. Some of them will push data on to the CAN bus at regular intervals like speed and RPM. On the other hand there are some other observations which are pushed on to the system only when they are required like door sensors in some vehicles.

In our model, we create a vector containing inputs from different systems. Each vector will then represent a single observation and our system will be trained for those observation sequences. We define the sequence of observations, $O = O_1, O_2, ...O_n$ where O_t is an observation vector at time t. Each observation vector $O_t = \{v_{t,1}, v_{t,2}, ...v_{t,n}\}$ where $v_{t,i}$ is the value of i^{th} component at time t. For example Speed is 20 mph, RPM is 3000, State of door is closed etc. are modeled as a single vector. During implementation, we interpret different values from particular slots in the CAN message and convert to decimal values before using them to train our model. To generate the HMM model we used "Statistics and Machine Learning toolkit" in Matlab. We use "hmmtrain" function to generate the model from the sequence of observations O. We chose to use Baum-Welch algorithm for training which will generate Transition and Emission Probabilities corresponding to test sequences.

VI. Anomaly Detection

We generated emission and transition probabilities for the HMM model using the data we collected from vehicles. In this phase we use this generated model to detect anomalies. By anomaly we mean a sudden deviation in the behavior of a vehicle interpreted from data on CAN bus. As we described earlier, we are not only detecting attack states, but also any unsafe or anomalous states. For example, even though it might not be caused by an attacker, opening of door at 200 mph is unsafe and hence we flag it. To detect unsafe

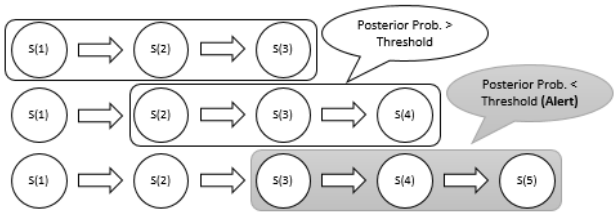

Fig. 3: Sliding Window For Anomaly Detection

states, we first convert the values from different components into a sequence of observation vectors in the same way as mentioned in section V. We then use a sliding window of "m" previous observations, $O_{window} = \{O_1, O_2, ...O_m\}$ as shown in Figure 3 to detect the presence of anomalies. The sliding window moves every time a new observation is available. One of the operations which we can do with HMM is to detect posterior probability of a given sequence. In this case, once the sliding window is determined, we use all observations in that window and determine the posterior probability of that sequence. As described before each of the observations would be a vector of different sensor values. It will generate a set of probabilities corresponding to each observation. If the probability of any such sequence is below a threshold, based on the generated model, it implies that probability of the current set of observations is very low and hence we identify it as an anomalous state.

We implemented our anomaly detection module using Matlab. The anomaly detection module has the model as its first input. In our implementation, the input stream from the CAN bus is fed to this module. The module will convert it into a sequences of observations using the same procedure we had used during model generation phase. Now when new observations are available, the module will pick up "m" previous observations from the sliding window and use "hmmdecode" from Matlab to find the posterior probability for that sequence in the window. Our module will now generate an alert, if the probability of any observation in the sequence is going below a set threshold value.

VII. Evaluation & Results

In order to evaluate our system, we need to verify that no alerts are generated during normal conditions and also that,

alerts are generated during unsafe conditions. To test normal conditions, we split collected data into two parts. The first part is used for training the model and the second part is used to verify if the model generates any false positives. For evaluating the system to detect unsafe states, we created different scenarios by injecting unsafe data into actual collected data. We had done a progressive evaluation scheme to test the performance of our model. Our first evaluation used only data from a single component. Further evaluations use more than one values at the same time. We describe our evaluation method and corresponding results below.

A. *Single Observation Evaluation*

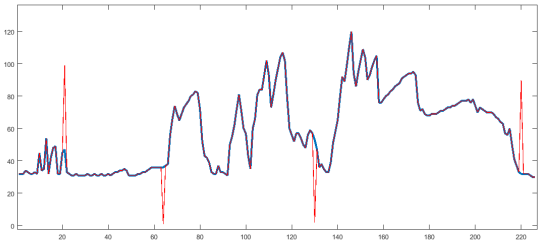

Fig. 4: Test Data for RPM as a single observation

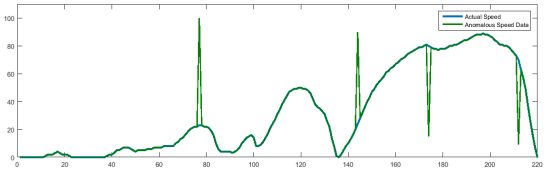

Fig. 5: Test Data for Speed as a single observation

No	Type of Change	Speed Alert Status	Result	RPM Alert Status	Result
1	⇑	False	✓	False	✓
2	⇓	False	✓	False	✓
3	⇑⇑	True	✓	True	✓
4	⇓⇓	True	✓	True	✓

TABLE I: Single Observation Evaluation. ⇑ - Gradual Increase, ⇑⇑ - Sudden Increase, ⇓ - Gradual Decrease ⇓⇓ - Sudden Decrease

We first trained our system only based on a single observed value. We used data from speed sensor and RPM sensor separately for this. Figure 5 represents a part of test data for speed shown graphically. Each of the spikes in it represents anomalous sudden change in speed caused as a result of the introduction of anomalous data to real data collected from vehicle. Ideally such a sudden spike is an unsafe state according to our hypothesis. Our generated model was able to detect each of those spikes. In order to make sure that this will work not only for that particular observation, we tried it with RPM sensor data shown in Figure 4. In a similar way the spike represents a very sudden change in RPM. We should note that the rate of change in RPM and Speed are different. RPM can increase more rapidly than speed. But since our model is based on real data collected from vehicles, it can detect all those variation which will normally happen in them. The results concluded from table I shows that anomalous changes, which cannot correspond to the normal context of a car were detected by our generated model.

B. *Multiple Observations Evaluation*

Since our model work well with single observations, we evaluated our technique considering multiple observations together in a vector. For this evaluation we chose the speed and RPM observations together. Both speed and RPM values are generated at regular intervals and hence we could map every speed value with an RPM value. A part of the anomalous values we generated and tested using our model is represented in figure 6. The spikes represent different anomalous situations which should not happen normally in a vehicle. We tested eight different anomalous situations in it. Each one represents either one or both of parameters being modified and can represent a potential malicious state. For example at time 118, the speed suddenly increases while the RPM value decrease, which is an anomalous scenario in a normal running vehicle. Similarly at time around 92 the RPM and speed increase abnormally. After generating the model, we tested these different cases and the evaluation results are described in table II. We can see that each of such situations were detected by our model and hence the results are promising. But we acknowledge the fact that we need to test our method with more anomalous states of varying degrees.

No	Speed	RPM	Alert Status	Result
1	⇑⇑	⇑⇑	True	✓
2	⇑⇑	⇓⇓	True	✓
3	⇓⇓	⇑⇑	True	✓
4	⇓⇓	⇓⇓	True	✓
5	⇑⇑	⟺	True	✓
6	⇓⇓	⟺	True	✓
7	⟺	⇑⇑	True	✓
8	⟺	⇓⇓	True	✓

TABLE II: Multiple Observation Evaluation. ⇑⇑ - Sudden Increase, ⇓⇓ - Sudden Decrease, ⟺ - Normal (from Test data)

VIII. Conclusion & Future Work

In this paper we introduced OBD_SecureAlert. It is a system which detects abnormal behavior in vehicles when they are being operated. Our model can be integrated with all current

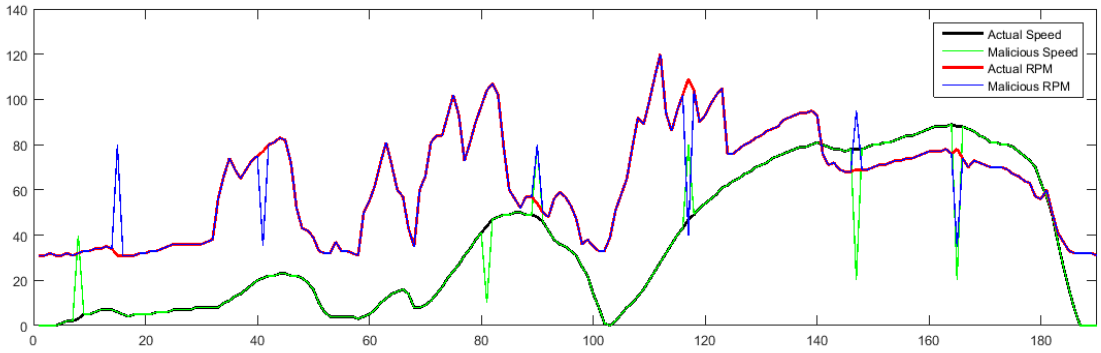

Fig. 6: Test Data for RPM and Speed together

and future car systems as a plug-n-play device or as a system module programmed on the on-board car computer. We successfully extract data from various manufacturers like Toyota, Honda and Chevrolet by attaching a device to their OBD port. Using the collected dataset, we generated a Hidden Markov Model for the prediction of anomalous states in vehicles. Our initial results show that such data analytic techniques could be successfully applied to identify anomalies and unsafe states in vehicles. Unlike some other methods, our method could successfully be utilized in both older and newer vehicles. We plan to extend our work by analyzing a real attack for further evaluations.

We can also further analyze the OBD data by applying other data mining algorithms on it. A logical next step is applying Conditional Random Fields and / or deep learning to gain better insight.

ACKNOWLEDGMENT

This work is done as a part of the Insure project sponsored by National Science Foundation (NSF). It is also partly supported by a Supplement from DoD to NSF award 1439663, and funds from the Oros Family Professorship. We thank Dr. Alan Sherman for his valuable comments and suggestions during this work. We also thank Michael R. Moore, Alan Barker and Joseph Raetano from Oak Ridge National Laboratory, for answering our various queries related to practical issues, we faced during this work.

REFERENCES

[1] U. S. D. of Transportation, *Household, Individual, and Vehicle Characteristics*, available at http://www.rita.dot.gov/bts/sites/rita.dot.gov.bts/files/publications/highlights_of_the_2001_national_household_travel_survey/html/section_01.html.

[2] A. Verbach, *The American Commuter Spends 38 Hours a Year Stuck in Traffic*, available at http://www.theatlantic.com/business/archive/2013/02/the-american-commuter-spends-38-hours-a-year-stuck-in-traffic/272905/.

[3] B. Semiconductors, *CAN with Flexible Data-Rate*, available at http://www.bosch-semiconductors.de/media/pdf_1/canliteratur/can_fd_spec.pdf.

[4] I. 11898-1:2003, *Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signalling*, available at http://www.iso.org/iso/catalogue_detail.htm?csnumber=33422.

[5] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham *et al.*, "Experimental security analysis of a modern automobile," in *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010, pp. 447–462.

[6] M. Wolf, A. Weimerskirch, and T. Wollinger, "State of the art: Embedding security in vehicles," *EURASIP Journal on Embedded Systems*, vol. 2007, no. 1, p. 074706, 2007.

[7] T. Hoppe, S. Kiltz, and J. Dittmann, "Security threats to automotive can networks–practical examples and selected short-term countermeasures," in *Computer Safety, Reliability, and Security*. Springer, 2008, pp. 235–248.

[8] T. Hoppe and J. Dittman, "Sniffing/replay attacks on can buses: A simulated attack on the electric window lift classified using an adapted cert taxonomy," in *Proceedings of the 2nd workshop on embedded systems security (WESS)*, 2007, pp. 1–6.

[9] C. Miller and C. Valasek, "Adventures in automotive networks and control units," in *DEF CON 21 Hacking Conference. Las Vegas, NV: DEF CON*, 2013.

[10] ——, "Remote exploitation of an unaltered passenger vehicle," Blackhat 2015, Tech. Rep.

[11] D. Storm, *Hack to steal cars with keyless ignition: Volkswagen spent 2 years hiding flaw*, available at http://www.computerworld.com/article/2971826/cybercrime-hacking/hack-to-steal-cars-with-keyless-ignition-volkswagen-spent-2-years-hiding-flaw.html.

[12] A. Hazem and H. A. Fahmy, "Lcap-a lightweight can authentication protocol for securing in-vehicle networks," in *10th escar Embedded Security in Cars Conference, Berlin, Germany*, vol. 6, 2012.

[13] A. Van Herrewege, D. Singelee, and I. Verbauwhede, "Canauth-a simple, backward compatible broadcast authentication protocol for can bus," in *ECRYPT Workshop on Lightweight Cryptography 2011*, 2011.

[14] B. Groza, S. Murvay, A. Van Herrewege, and I. Verbauwhede, "Libracan: Lightweight broadcast authentication for controller area networks."

[15] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno *et al.*, "Comprehensive experimental analyses of automotive attack surfaces." in *USENIX Security Symposium*. San Francisco, 2011.

[16] M. Ruta, F. Scioscia, F. Gramegna, and E. Di Sciascio, "A mobile knowledge-based system for on-board diagnostics and car driving assistance," in *UBICOMM 2010, The Fourth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*. Citeseer, 2010, pp. 91–96.

[17] Z. Bar-Joseph, "Analyzing time series gene expression data," *Bioinformatics*, vol. 20, no. 16, pp. 2493–2503, 2004.

[18] Y. Zhang, "Prediction of financial time series with hidden markov models," Ph.D. dissertation, Simon Fraser University.

[19] Y. Guo, G. Poulton, P. Corke, G. Bishop-Hurley, T. Wark, and D. L. Swain, "Using accelerometer, high sample rate gps and magnetometer data to develop a cattle movement and behaviour model," *Ecological Modelling*, vol. 220, no. 17, pp. 2068–2075, 2009.