CarLab: Framework for Vehicular Data Collection and Processing

Mert D. Pesé Arun Ganesan Kang G. Shin
The University of Michigan – Ann Arbor
{mpese,arungan,kgshin}@umich.edu

ABSTRACT

Due to the growth of intelligent and self-driving vehicles, there are a multitude of data-driven applications such as user monitoring or traffic modeling and control. Each application often uses its own data-collection platform, leading to a scattered landscape of solutions for vehicular data-driven research and app development. We propose CarLab, a flexible and open vehicular data-collection platform which unifies this landscape of vehicular data-driven research and app development.

In this paper, we survey the field of vehicular data collection, describe the system architecture of CarLab and related research issues.

KEYWORDS

Vehicular Data Collection; System Design; Security; Privacy; V2X

1 INTRODUCTION

The automotive industry is experiencing a major change as vehicles are becoming part of the Internet. As a result of this paradigm shift, vehicles can no longer be viewed as purely mechanical platforms, but are smart cyber-physical systems with highly sophisticated Electric/Electronic (E/E) architectures and a large amount of data exchanged between Electronic Control Units (ECUs) on several In-vehicle Networks (IVNs). There have been several applications in academic [18, 26, 35] and commercial domains [4, 9, 10] which leverage this large influx of data. A recent CNN article [25] argued that data collected from self-driving cars is very valuable, and we need more intelligent data collection to sift through the massive amounts of data and selectively transmit the important bits.

To address this growing need for data collection, most OEMs and suppliers offer their own hardware (usually based on the generic ELM327 OBD-II to Bluetooth chip) and smartphone app as a standalone solution. In addition, OEMs have begun to offer proprietary platforms which can integrate several apps and give them access to user data without using third-party interfacing tools. Examples are BMW's Connected Drive [2], Audi Connect [1] or GM's OnStar [5] to just name a few. However, these platforms are either integrated within the car, or require different apps for different car makers. Academic efforts for data collection face a similar isolation where each data-collection effort involves a custom platform developed by the academic researchers. For example, MIT's CarTel [22] project, UMass' DOME [32] project, University of Michigan's SafetyPilot [13] all use custom data-collection platforms even though they perform similar functions. By requiring a new platform tailored to each project, it increases the barrier for entry in vehicle research and stifles innovation of vehicle app developers.

To mitigate the above-mentioned problems, we introduce *CarLab*, an open-source vehicular data-collection platform. It provides a common base for varieties of vehicular apps, thereby unifying the isolated solutions that exist today. Developers and researchers can easily design apps which reside on top of our CarLab platform. It distinguishes itself from existing solutions in the following four areas.

- **Diverse Hardware Support.** CarLab provides a hardware abstraction layer to interface with multiple data-collection interfaces, including OBD-II, CAN bus, smartphone/watch sensors, wearable sensors, and Internet-based data sources. Existing solutions only support limited hardware and are not flexible to add new hardware once the data-collection subsystem has been deployed.
- **Developer API.** Developers and researchers can write apps for CarLab's base platform using a user-friendly API. Apps written for CarLab are partial applications which run inside the CarLab ecosystem, making it easy for developers to write apps for CarLab. This is distinguished from existing work which either does not support third party developers, or requires that they develop full apps to interface with their API.
- **Secure and Private Data Collection.** CarLab provides a variety of security and privacy features to protect the end-users and ensure the integrity of the data collected by apps running on the CarLab platform. We enforce a permission model to ensure the data collected using each app cannot reveal the users' identities without their explicit permission. Furthermore, we employ cryptography to encrypt all data transfers to ensure confidentiality and integrity.
- **Real-time Network Communication.** CarLab flexibly communicates with the cloud using available network such as WiFi or LTE/5G. We balance the needs of the apps with network availability/condition to schedule data transfers.

To the best of our knowledge, there is no such open data-collection platform. We envision that CarLab will fill this need for a unified and flexible data-collection platform, and engender an uprising of useful vehicular apps. The rest of the paper is organized as follows. *Sec. 2* gives an overview of related work, while *Sec. 3* introduces the general system architecture of the CarLab platform. *Sec. 4* details the research issues related to CarLab and *Sec. 5* concludes the paper.

ACM CarSys'17, Snowbird, Utah, USA
 978-x-xxxx-xxxx-x/YY/MM…$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

	Real-Time Network Comm.	Diverse Hardware Support	Flexible Developer API	Secure and Private Data Collection
Safety Pilot [13]	✓	✓		
IVBSS [27]		✓		
CANOPNR [16, 17]	✓			
SMaRTCaR [14]	✓	✓		
Bender et al. [12]	✓	✓		
CarTel [18, 22, 28]		✓		
BMW CARDATA [3]	✓		✓	✓
Automatic [4]	✓		✓	
Torque [10]	✓		✓	
CarLab	✓	✓	✓	✓

Table 1: Comparison of vehicular data collection and processing projects. Many existing data-collection efforts support real-time network communication using WiFi, LTE or DSRC. Depending on the project they are used for, they also support multiple hardware sources. However, very few of them support an open and flexible developer API so that others can utilize the platform. Although they do keep all data confidential, they do not take the user's privacy into account in designing the platform. For example, if the servers were compromised, an attacker may be able to de-anonymize the end-users.

2 RELATED WORK

There have been diverse efforts on the different aspects of vehicular data collection as presented below.

Vehicular Data Use-Cases. Vehicular data is used in a variety of domains, such as driver drowsiness monitoring [30, 34], driving pattern learning and suggestions [21, 26], and city-scale modeling [11, 18, 23]. The abundant research in this field is an indicator of the importance/value of vehicular data collection, although each data-collection effort is isolated and requires attracting participants, developing and distributing the platform.

Data-Collection Efforts. The authors of [29] briefly introduced vehicle data collection using an OBD-II dongle. They do not leverage the use of smartphone sensors, but only collect the data from the vehicle and stream it in JSON format over the phone which acts as a gateway to the remote cloud where it is stored in a non-relational database and visualized. They do not encrypt any data either.

MIT's CarTel project developed and installed a portable data-collection platform in many vehicles [22]. It includes GPS, an OBD reader, and wireless interfaces to transmit the collected data back to the server. This platform was used for a variety of vehicular research projects such as pothole monitoring [18], network behavior [28], and traffic and location privacy [35]. Although their data-collection platform was used for multiple projects, it lacks the flexibility CarLab provides. Projects using CarLab either used existing traces [28] or installed additional sensors such as an accelerometer [18]. The description of the CarLab platform [22] doesn't allow for re-utilizing the same infrastructure for new projects.

Researchers at UMass conducted a multi-year data-collection effort called *Diverse Outdoor Mobile Environment* (DOME) to study network characteristics in the Amherst area [32]. Their testbed was installed inside 40 UMass buses and used to study WiFi, 3G, and GPRS connectivity. In contrast to their testbed, CarLab integrates multiple sensors in a unified view of the vehicle and the driver. Furthermore, CarLab is designed to support third-party apps which keep the testbed open to innovative apps and research directions.

Private and Secure Data Collection. Data collection enforcing security and preserving the privacy of the driver is another important issue. Li *et al.* [24] designed a framework for securely collecting data from vehicle and mobile sensors and transferring them to the cloud using fine-grained and context-aware uploading policies which enforce privacy requirements of the driver's data by encrypting or dropping highly sensitive data which could be used to determine the driver's identity.

Fawaz and Shin [19] modeled location leakage of different apps residing on the smartphone. Similar to their model, CarLab monitors the leakage of private information to each app running on top of CarLab.

Felt *et al.* [20] described guidelines for designing permission models for smartphone ecosystems. They concluded that one permission model may be insufficient for different needs of a smartphone. Using their work on permission and usable privacy, we explore different permission models for apps residing on CarLab.

3 SYSTEM ARCHITECTURE

We now briefly introduce the system architecture of the CarLab project in which the data-collection platform is embedded.

3.1 Overview

CarLab intends to offer a highly modularizable platform to collect and process data from vehicles and mobile devices carried by the driver and passengers. Its ultimate goal is to offer an entire ecosystem for vehicular applications which can rely on a secure and optimized architecture which is kept as generic as possible and can thus be tailored to the user's needs. The target users consist of researchers as well as regular drivers and app developers. The number of researchers involved in analyzing vehicular and mobile sensor data has been constantly rising, as seen in recent literature (see Sec. 2). CarLab intends to offer them an open-source and highly modularized architecture which they can deploy for their research and tailor it to their needs. Apart from re-usability of code for various possible projects, researchers can possibly contribute to

the current architecture by adding modules to the layers. Since the platform is open-source and offers a good base for further research, we expect CarLab to continuously evolve and benefit everyone deploying it. Furthermore, CarLab is targeted for deployment by regular drivers who are interested in analytics of their data. As we will describe in the next subsections, users of this platform just need a simple hardware interface to connect to the vehicle as well as a smartphone which runs CarLab and communicates with both the vehicle and the backend. Finally, developers can submit their apps to CarLab's app store which offers permission management very much like in Android. App developers do not have to worry about any layers below their application layer any more since data collection is being handled by CarLab. Furthermore, third-party apps usually come with proprietary hardware dongles. A different app might require a different dongle which both increases the cost for the user as well as has technical limitations (Y-Cable to split the OBD-II data is needed). Multiple apps can reside within CarLab and interface with the vehicle through one generic dongle.

An overview of CarLab's system architecture is given in Figure 1. In what follows, we will elaborate on all layers to better understand the interconnection of the single components. Note that our focus in this paper is placed on the introduction of the two bottom layers as they will be especially relevant for the data-collection phase of CarLab.

3.2 Hardware Interfaces

Unlike other existing commercial solutions each of which usually operates on particular hardware, CarLab works with several hardware interfaces via a Hardware Abstraction Layer (HAL). There are multiple interfaces available to access the vehicle's network. Vehicular data is collected from a set of ECUs of the vehicle. These devices are usually interconnected with each other by an on-board communication bus, or IVN. Although Electric/Electronic (E/E) architectures differ between cars, an IVN usually consists of multiple bus systems using different protocols and physical layers which all converge in a node. This node is called Central Gateway (CGW) and can be regarded as a router between the different bus segments. Bus systems are segmented with ECUs according to the ECU's purpose, mostly due to timing constraints. The most common bus found in vehicles is Controller Area Network (CAN). For instance, ECUs for engine, transmission and braking functions are located on the power-train CAN. Other commonly used CAN bus lines are the Comfort CAN, Body CAN, and Infotainment CAN.

The major point of entry into a vehicle is the on-board diagnostics (On-board Diagnostics II (OBD-II)) interface. This connector is mandatory for all vehicles sold in the US since 1996, and it is used for emissions measurements and diagnostic features. This connector lies on the Diagnostic CAN which is connected to the CGW and can thus send and receive CAN messages to and from all other CAN buses as long as the CGW routes them through. Since CarLab intends to offer an easy plug-and-play solution for everyone, without having to tap into the IVN wires, the OBD-II port will be the supported point of entry into the vehicle.

Various hardware interfaces exist which can be plugged into this port. The most popular solutions are Bluetooth and WiFi dongles which are based on the ELM327 chip and can be directly connected to a phone. Furthermore, there are more professionally wired solutions which can be connected to a laptop via USB. Although they have a performance advantage over wireless dongles, they mostly have similar functionality to them. Together with the usability requirement in CarLab, we will only support these wireless dongles. Note that several dongles from different manufacturers exist as depicted in Figure 1 [6, 8]. Since the Society of Automotive Engineers (SAE) is planning to harden or eventually close the OBD-II port during vehicular motion in the near future [31], our platform must also be ready for future developments. Using our modular architecture, we can provide support for other ways than the OBD-II port to access the IVN, such as obtaining the data directly from the Original Equipment Manufacturer (OEM). As mentioned before, our platform also uses Vehicle-to-vehicle (V2V) communication to relay data to other vehicles in certain cases as we will explain later.

Finally, smartphones and wearables are also part of this layer. Since mobile sensor data such as acceleration and GPS or heartrate can be used in various applications, we want to leverage their availability in our CarLab platform. CarLab is designed to collect smart-phone/watch sensors from both drivers and passengers. Additionally, CarLab collects web-based information such as the current weather, or the local traffic conditions.

3.3 Hardware Abstraction Layer (HAL)

The Hardware Abstraction Layer (HAL) is used to offer support to different hardware interfaces. At present, all dongles support both the OBD-II and CAN protocol. Due to different implementations, different dongles can result in a different streaming format of protocol data. Through hardware-specific plugins for each dongle, the HAL returns a standardized API to the upper layers so that app developers do not have to deal with a specific protocol format.

The OBD-II protocol was originally designed for emission-relevant information only, but now it provides extensive engine control, chassis, body, and comfort function diagnostic information. Information about several standardized parameter ID (PID) called OBD Parameter IDs (OBD-II PIDs) [7] can be obtained without prior knowledge of the vehicle architecture or its message format. The procedure for obtaining information about OBD-II PIDs is similar to a publish-subscribe mechanism. Depending on the sampling frequency constraints of the interface, the monitor entity can subscribe to the relevant ECUs by sending periodic requests for its desired set of OBD-II PIDs. The respective ECUs will then respond within a certain time.

The second supported protocol at this point of time is CAN. ECUs broadcast their messages on this network so everyone connected to it can read and write. The relevant part of a CAN message consists of a CAN ID and an 8 byte payload. For CAN packets, each ECU determines itself if it is interested in a message with that specific CAN ID. Note that the CAN ID does not reveal any information about the sender of the message. Using direct access to the CAN bus could increase the sampling frequency for the data points, but requires knowledge of the CAN IDs of the individual messages. This is proprietary information and can be either requested from the OEM or reverse engineered [15, 33]. App developers have to be aware of this constraint. Nevertheless, proprietary dongles/gateways such as Ford's OpenXC platform [8] can help circumvent this problem.

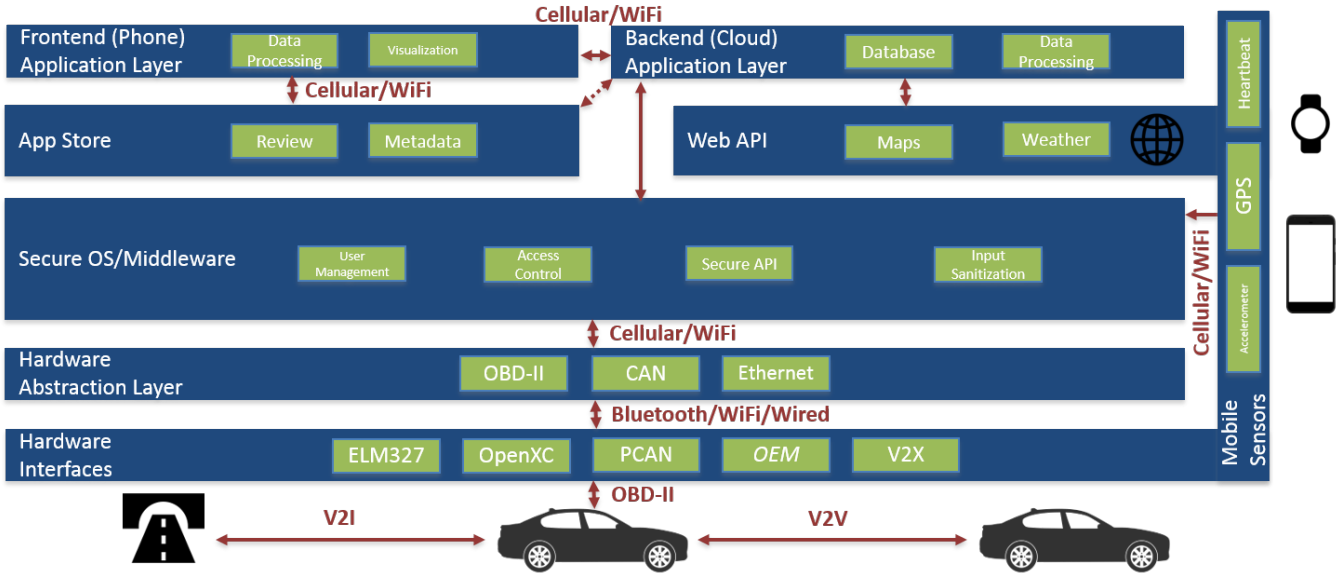

Figure 1: CarLab Architecture

3.4 Secure Middleware

This layer connects the hardware layers with the application layer and builds the core of our platform. The middleware has multiple tasks which are summarized as follows.

- **Permission model:** Apps have a permission model such as in Android. Not every app needs to read the entire data from the IVN. For instance, if OBD-II is used as the underlying protocol, only certain OBD-II PIDs will be allowed to be accessed by an app. Except OBD-II PIDs requests, there is also no need for regular end-users to possess write access to the IVN. This access control method has two major benefits. First, it offloads traffic and saves bandwidth which will be an important factor for optimized data collection. Second, it is necessary for security and privacy of the data and vehicle. Apps which are exploiting write access to the IVN can very easily affect the functional safety of the vehicle. Apps which are allowed to read the entire bus can exploit the user's data and draw conclusions regarding its identity.
- **User Management:** Since CarLab will be used by multiple people, a proper user management has to be supported. This is a very sensitive subject since privacy leakage has to be stopped. As some researchers at the University of Michigan-Dearborn pointed out [24], vehicular and mobile data can be used to determine the identity of its driver. Since CarLab will support a granular permission model as mentioned above, users will be allowed to choose what parts of their data will be allowed to be used by apps. This information has to be transmitted and stored securely on the backend. CarLab users have to create a profile at the first start and log in and authenticate every time they restart the app. Due to a different permission model, it will also be supported to distinguish between researchers, app developers and regular end-users.
- **Input Sanitization:** We believe that many apps will heavily rely on mobile sensor data such as GPS. Since smartphones and wearables introduce an additional attack vector and are more likely to be tampered with than vehicular sensors, the input from these sources have to be sanitized against possible spoofing. This can be achieved by checking it against similar redundant sensor data as well as historical contexts. This will be a crucial part of the secure middleware to guarantee a correct functioning of the CarLab eco-system.
- **Secure API:** The middleware offers an API for third-party app developers to interface the vehicle as well as mobile sensors. API calls will be made secure in order to guarantee integrity of data and prevent eavesdropping on the wireless links. Besides standardizing the access to the lower layers of CarLab, a carefully designed API will also prevent any unwanted calls which might damage the circumvent the security mechanisms of the platform.

3.5 Application Layer

CarLab will support developers to integrate their apps into the existing eco-system. Besides the API which will be offered to them through the middleware as mentioned above, they will be given access to the backend where they can run the server-side of their apps. The application layer will consist of a frontend and a backend. App developers have to submit code for the CarLab app frontend and also for the server backend.

3.6 App Store

The main idea behind the CarLab app store is that app developers and researchers can offer their apps to their target users through an easy-to-use app store which is embedded inside the CarLab app. Every app has to specify certain metadata before it can be submitted and published in the app store. This metadata contains the app permissions which are governed by the middleware as well as pre-defined app requirements which are necessary for optimizing the data collection. These requirements comprise latency, storage, computational power. More details about how these requirements can be used for efficient data collection will be provided in the next subsection.

4 RESEARCH ISSUES

4.1 Efficient Data Collection

At the bottom of the CarLab architecture, there is the need for a secure and optimized data collection from vehicles as well as mobile sensors. Although there are several papers which deal with this, none of them focuses on the data-collection process itself. Since the focus of these researchers is on the application based on the data collection, they simply choose to log the data and upload it when WiFi is available. Different applications have different requirements towards parameters such as end-to-end (e2e) latency, storage, computation resources, etc. For instance, an intrusion/collision monitoring app relies on real-time data to detect and prevent possible intrusions/collisions to the IVN. Other apps can perform big data analytics offline after the data has been uploaded to the cloud. Similarly, apps differ in how much data is stored and in the amount of time and CPU or GPU power needed for processing. As a result of app's varying requirements towards the data collection and processing system, CarLab will adapt flexibly to these parameters and find the optimal way to satisfy those requirements.

Based on these parameters defined by app developers, user preference metrics such as maximal cellular bandwidth cost and dynamic parameters such as network connectivity, CarLab calculates an optimal path to collect and process the data. These paths are depicted in Figure 2. Collected data can be processed locally on the smartphone which acts as a gateway to the cloud or can be shunted to the backend. Local processing makes sense in a scenario with real-time requirement and low need for computational resources whereas cloud processing is suitable for performance-hungry algorithms with no real-time requirement, such as offline data analytics. A third possible path besides local and cloud processing could be relaying data over another vehicle to the cloud by leveraging V2V communication. This will be elaborated on in the next subsection.

4.2 Leveraging V2X Communication

Vehicle-to-everything (V2X) communication is becoming more popular as part of connected and autonomous vehicles. The term V2X includes vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) as well as some other types of communications. Connected cars exchange large amounts of data, and thus the cloud with high storage capabilities will be required to store and process the data [36]. V2X communication can be leveraged inside CarLab for multiple tasks. Some of them will be discussed below.

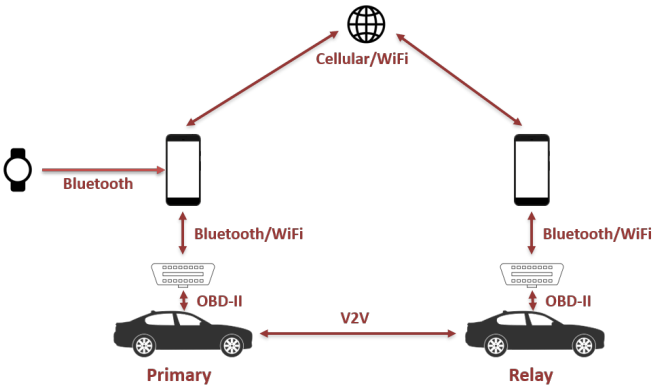

Figure 2: Possible paths to collect and process data

The basic idea behind V2V is that vehicles can talk to each other and inform others about their physical parameters, primarily in order to enhance safety. Another possibility to use V2V communication is to use other vehicles around as a relay for their collected data. For instance, a better cellular signal quality of a different CarLab user in a nearby vehicle can be capitalized to upload real-time data to the cloud. Potential research questions are how this additional link will affect latency or how privacy of the data can be preserved on a stranger's phone. Furthermore, the CarLab peer has to be rewarded for sharing his mobile bandwidth with others during upload.

Finally, V2I communication forms another valuable data source which can be offered to app developers through the API.

4.3 Security and Privacy Research

Designing a framework such as CarLab imposes several security and privacy questions. As already mentioned in Section 3.4, a secure middleware between application and hardware layers is inevitable. Key research questions will be how and what part of data will be encrypted. Since key management and encryption will also affect parameters of the data-collection process such as latency, its effect has to be investigated thoroughly. A representative adversary model has to be defined and possible attacks have to be studied in order to consolidate the security layer and mitigate the effect of any malicious attack towards the CarLab platform.

5 CONCLUSION

CarLab is an open and flexible vehicular data-collection platform with 4 distinguishing features from other data-collection efforts. It offers (1) real-time and opportunistic network connectivity, which allows apps to choose between networks to balance cost and urgency, (2) diverse hardware support through a hardware abstraction layer, (3) flexible developer API which allows third-party developers and researchers to quickly program for, and utilize the services of CarLab, and (4) secure and private data collection which protects the user's privacy from curious third-party apps. CarLab unifies the currently scattered vehicular data-collection ecosystem. This unification will stimulate proliferation of innovative apps and research efforts in vehicular data collection and applications.

ACKNOWLEDGEMENTS

The work reported in this paper was supported in part by the NSF under Grant CNS-1505785 and an Intel Labs contract, as well as a University Michigan MCity grant. Victor Boyse-Peacor's help in CarLab development is also gratefully acknowledged.

REFERENCES

[1] Audi connect — audi usa. https://www.audiusa.com/technology/intelligence/audi-connect, 2017.
[2] Bmw connected drive. https://www.bmw.com/en/topics/fascination-bmw/connected-drive/overview.html, 2017.
[3] Bmw connecteddrive: Bmw cardata. https://www.bmw.com/en/topics/fascination-bmw/connected-drive/bmw-cardata.html, 2017.
[4] Connect your car to your digital life with automatic. https://www.automatic.com/, 2017.
[5] Gm onstar. https://www.onstar.com/us/en/home.html, 2017.
[6] Obd - elm electronics. https://www.elmelectronics.com/products/ics/obd/, 2017.
[7] Obd-ii pids, the free encyclopedia, August 2017. wikipedia.org [Online; posted August-7-2017].
[8] The openxc platform. http://openxcplatform.comOpenXC, 2017.
[9] Snapshot from progressive — big discounts for good drivers. https://www.progressive.com/auto/discounts/snapshot/, 2017.
[10] Torque — obd2 performance and diagnostics for your vehicle. https://torque-bhp.com/, 2017.
[11] Agamennoni, G., Ward, J. R., Worrall, S., and Nebot, E. M. Anomaly detection in driving behaviour by road profiling. In *Intelligent Vehicles Symposium (IV), 2013 IEEE* (2013), IEEE, pp. 25–30.
[12] Bender, A., Ward, J. R., Worrall, S., Moreyra, M. L., Konrad, S. G., Masson, F., and Nebot, E. M. A flexible system architecture for acquisition and storage of naturalistic driving data. *IEEE Transactions on Intelligent Transportation Systems 17*, 6 (June 2016), 1748–1761.
[13] Bezzina, D., and Sayer, J. Safety pilot model deployment: Test conductor team report. Tech. rep., DOT, 06 2015.
[14] Campolo, C., Iera, A., Molinaro, A., Paratore, S. Y., and Ruggeri, G. Smartcar: An integrated smartphone-based platform to support traffic management applications. In *2012 First International Workshop on Vehicular Traffic Management for Smart Cities (VTM)* (Nov 2012), pp. 1–6.
[15] Correa, C. Techniques for learning a vehicle's can database, June 2010. http://www.ukintpress-conferences.com [Online; posted June-24-2010].
[16] Enriquez, D., Jenson, S., Bautista, A., Hawn, P., Kim, S.-i., Ali, M., and Miller, J. On software-based remote vehicle monitoring for detection and mapping of slippery road sections. *International journal of intelligent transportation systems research 15*, 3 (2017), 141–154.
[17] Enriquez, D. J., Bautista, A., Field, P., i. Kim, S., Jensen, S., Ali, M., and Miller, J. Canopnr: Can-obd programmable-expandable network-enabled reader for real-time tracking of slippery road conditions using vehicular parameters. In *2012 15th International IEEE Conference on Intelligent Transportation Systems* (Sept 2012), pp. 260–264.
[18] Eriksson, J., Girod, L., Hull, B., Newton, R., Madden, S., and Balakrishnan, H. The pothole patrol: using a mobile sensor network for road surface monitoring. In *Proceedings of the 6th international conference on Mobile systems, applications, and services* (2008), ACM, pp. 29–39.
[19] Fawaz, K., and Shin, K. G. Location privacy protection for smartphone users. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* (2014), ACM, pp. 239–250.
[20] Felt, A. P., Egelman, S., Finifter, M., Akhawe, D., Wagner, D., et al. How to ask for permission. In *HotSec* (2012).
[21] Ganti, R. K., Pham, N., Ahmadi, H., Nangia, S., and Abdelzaher, T. F. Greengps: a participatory sensing fuel-efficient maps application. In *Proceedings of the 8th international conference on Mobile systems, applications, and services* (2010), ACM, pp. 151–164.
[22] Hull, B., Bychkovsky, V., Zhang, Y., Chen, K., Goraczko, M., Miu, A., Shih, E., Balakrishnan, H., and Madden, S. Cartel: a distributed mobile sensor computing system. In *Proceedings of the 4th international conference on Embedded networked sensor systems* (2006), ACM, pp. 125–138.
[23] Jiang, B., and Fei, Y. Traffic and vehicle speed prediction with neural network and hidden markov model in vehicular networks. In *Intelligent Vehicles Symposium (IV), 2015 IEEE* (2015), IEEE, pp. 1082–1087.
[24] Li, H., Ma, D., Medjahed, B., Wang, Q., Kim, Y. S., and Mitra, P. Secure and privacy-preserving data collection mechanisms for connected vehicles. Tech. rep., SAE Technical Paper, 2017.
[25] McFarland, M. Your car's data may soon be more valuable than the car itself, February 2017. money.cnn.com [Online; posted February-7-2017].
[26] Meseguer, J. E., Calafate, C. T., Cano, J. C., and Manzoni, P. Drivingstyles: A smartphone application to assess driver behavior. In *Computers and Communications (ISCC), 2013 IEEE Symposium on* (2013), IEEE, pp. 000535–000540.
[27] P. Green, J. M. Sullivan, O. T. J. O. M. L. B. J. D. J. S. E. B., and Sayer, J. Integrated vehicle-based safety systems (IVBSS): Human factors and driver-vehicle interface (DVI) summary report. Tech. rep., 2008.
[28] Ravindranath, L., Newport, C., Balakrishnan, H., and Madden, S. Improving wireless network performance using sensor hints. In *Proceedings of the 8th USENIX conference on Networked systems design and implementation* (2011), pp. 21–21.
[29] Reininger, M., Miller, S., Zhuang, Y., and Cappos, J. A first look at vehicle data collection via smartphone sensors. In *Sensors Applications Symposium (SAS), 2015 IEEE* (2015), IEEE, pp. 1–6.
[30] Sahayadhas, A., Sundaraj, K., and Murugappan, M. Detecting driver drowsiness based on sensors: a review. *Sensors 12*, 12 (2012), 16937–16953.
[31] Shuttleworth, J. Sharpening the focus on obd-ii security, February 2017. sae.org [Online; posted February-7-2017].
[32] Soroush, H., Banerjee, N., Balasubramanian, A., Corner, M. D., Levine, B. N., and Lynn, B. Dome: a diverse outdoor mobile testbed. In *Proceedings of the 1st ACM International Workshop on Hot Topics of Planet-Scale Mobility Measurements* (2009), ACM, p. 2.
[33] Staggs, J. How to hack your mini cooper: reverse engineering can messages on passenger automobiles. *Institute for Information Security* (2013).
[34] Teyeb, I., Jemai, O., Zaied, M., and Amar, C. B. A drowsy driver detection system based on a new method of head posture estimation. In *International Conference on Intelligent Data Engineering and Automated Learning* (2014), Springer, pp. 362–369.
[35] Thiagarajan, A., Ravindranath, L., LaCurts, K., Madden, S., Balakrishnan, H., Toledo, S., and Eriksson, J. Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems* (2009), ACM, pp. 85–98.
[36] Wollschlaeger, I. D. Whatfis next? v2v (vehicle-to-vehicle) communication with connected cars. https://www.wired.com/insights/2014/09/connected-cars/, Aug 2015.