IoT Security Development Framework for Building Trustworthy Smart Car Services

Jesus Pacheco, Shalaka Satam, Salim Hariri
Electrical and Computer Engineering Department
The University of Arizona
Tucson, Arizona, USA
{jpacheco, shalakasatam, hariri}@email.arizona.edu

Clarisa Grijalva, Helena Berkenbrock
Global Initiatives
The University of Arizona
Tucson, Arizona, USA
{clarisagl, hberkenbrockniem}@email.arizona.edu

Abstract— **The Internet of Things (IoT) will connect not only computers and mobile devices, but it will also interconnect Smart cars, buildings, homes, and cities, as well as electrical grids, gas, and water networks, automobiles, airplanes, etc. However, with the introduction of IoT, we will be experiencing grand challenges to secure and protect its advanced information services due to the significant increase of the attack surface, complexity, heterogeneity and number of interconnected resources. In this paper, we present an IoT Security Development Framework (ISDF) to build trustworthy and highly secure applications and services. The ISDF enables developers to consider security issues at all IoT layers and integrate security algorithms with the functions and services offered in each layer rather than considering security in an ad-hoc and after thought manner. We also show how this framework can be used to develop highly secure and trustworthy Vehicle Information and management Portal (VIMP) services and how to apply our Anomaly Behavior Analysis (ABA) methodology to secure and protect these services against any type of attacks.**

Keywords—smart car; autonomous vehicle; internet of things; threat model; control area network; cyber security

I. Introduction

Advances in mobile and pervasive computing, social network technologies and the exponential growth in Internet applications and services will lead to the development of the next generation of Internet (Internet of Things, IoT) that are pervasive, ubiquitous, and touch all aspects of our life [1]. However, the integration of physical and cyber systems will dramatically increase the vulnerability of interdependent systems [2]. In this paper, we focus on one emerging IoT service associated with smart and autonomous cars that will have major security problems. Modern and soon smart vehicles are controlled by complex distributed systems comprising large amount of heterogeneous nodes with rich connectivity provided by internal networks and Internet. With such exponential increase in vehicle intelligence and connectivity, security and privacy have become the main concerns for automotive systems [3]. Researchers have shown that by compromising a single control unit, a capable attacker may gain access to other vehicle units via internal communication buses such as controller area network (CAN), and attack critical subsystems [3]. As CAN gets interconnected with IoT resources and services, it becomes easy targets to cyber adversaries, especially since it was never designed to handle cyber threats. This makes CAN data vulnerable to falsification attacks that lead to incorrect information delivery to users [3], and thus causing them to take wrong and dangerous actions or to be unaware of an ongoing attack as was the case in Stuxnet attack [4]. It also allows adversaries to potentially execute malicious commands on control systems, causing harmful actions (e.g. Disable brake system) [5]. Therefore, it is critically important to secure and protect smart vehicle operations against any type of cyber-attacks.

In this paper, we first introduce an IoT hierarchical architecture that can be used to develop IoT applications. We then extend that architecture to the IoT Security Development Framework (ISDF). The objective the ISDF is to enable developers to address security issues in a systematic way while designing and developing each IoT layer rather than considering security as an afterthought and in ad-hoc manner. In our approach, IoT hierarchy consists of four layers: Application, Service, Communications and End-Devices layers. By insuring for each layer that all existing vulnerabilities and threats can be identified and mitigation solutions will be applied, the ISDF will provide the architectural support to deliver trustworthy IoT services that can: 1) Protect IoT services against epidemic attacks; 2) Ensure that critical IoT systems can survive faults and destructive attacks; and 3) Ensure IoT security and privacy. We show how to apply this framework to develop a trustworthy Vehicle Information and Management Portal (VIMP) services to support smart car applications. By connecting cars to VIMP services, we can offer revolutionary new services in entertainment, communication, collaboration, and on-line monitoring to increase safety by proactively and reactively warning about the vehicle current dangerous conditions, continuous access to field data, and on-line firmware update, just to name a few. In addition, show how our methodology can be applied to secure and protect the VIMP services against a wide range of cyber-attacks that target vehicle sensors. We have evaluated our approach by launching several cyberattacks (e.g. Replay, and Flooding attacks) against our Smart Vehicle (SV) testbed. The results show that our ISDF can be used to implement security mechanisms to protect the normal operations of each IoT layer.

The rest of the paper is organized as follows. In section II we provide background information about CAN, SVs cyber security issues, IoT cyber security, and ABA-IDS. In Section III, we explain our IoT architecture for smart car services. In section IV, we present an IoT security development framework

978-1-5090-3865-7/16/$31.00 ©2016 IEEE

(ISDF) for building smart car applications. Section V is devoted to describe our Vehicle Information and Management Portal (VIMP) development and its potential services. In Section VI, our smart vehicle testbed and aba-ids methodology are discussed. In section VII we present our experimental environment and evaluation results. In section VIII, we conclude the paper and discuss future research.

II. BACKGROUND

A. Car Area Network (CAN)

CAN is a protocol for short messages, making it suitable for transmission of trigger signals and measurement values, it efficiently supports distributed control systems [6]. The physical layer of the model defines the communication between devices. The ISO architecture for CAN is shown in Figure 1.

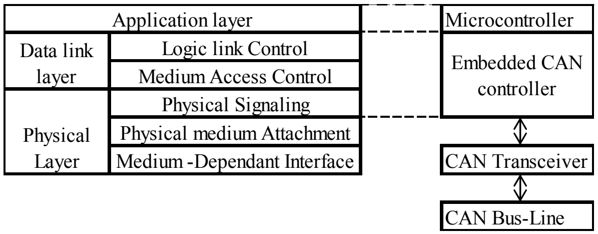

Fig. 1. CAN architecture

Major faults affecting CAN based networks are [6]: 1) Node Stuck-at-bit fault. In this state a faulty node will issue a constant bit value preventing the rest of the participating nodes from communicating. 2) Medium Break-up fault: This situation happens when the CAN bus is split into 2 or more sections (e.g. because of cable damage), disrupting the communication. 3) Babbling Idiot fault. It occurs when a node occupies the CAN bus by constantly sending high-priority messages.

B. Smart Vehicles Cyber Security

Modern vehicles and autonomous cars are controlled by complex distributed systems comprising several nodes with rich connectivity provided by internal networks [5]. While this structure is efficient, it also creates significant vulnerabilities for cyberattacks. For example, an attacker connected to a vehicle's internal network can invade control systems, including safety critical elements such as brakes. Researchers have shown that modern vehicles can be attacked from a variety of interfaces including physical access such USB, and remote access such as tire pressure sensors [3]. Attacks on SVs can be classified into [5]: indirect physical access, short-range wireless access, and long-range wireless access. A special category of SV is the autonomous car (AC). ACs possess the ability of self-driving by using five basic functions:1) Perception to sense the surrounding environment, 2) localization to find the position of the AC in a map, 3) planning to determine the behavior and motion of the AC based on the information from perception and localization, 4) control to guide the car by following the planning function, and 5) system management to supervise the overall autonomous driving system [3]. Since there is no human-taken decisions for ACs, reliability and security are crucial components [6].

C. IoT Cyber Security

IoT can be viewed as a ubiquitous network that enables monitoring and controlling a large number of heterogeneous devices that are geographically dispersed by collecting, and processing and acting on the data generated by smart objects [7][8]. Traditional IT security solutions are not directly applicable to IoT due to the following issues [7]: 1) The IoT extends the "internet" through the traditional internet, mobile network, non IP networks, and cloud computing; 2) Computing platforms are constrained in memory and processing capability, consequently they may not support complex security algorithms; 3) All "things" will communicate with each other. This leads to multiple entry points that can be used to exploit existing vulnerabilities; and 4) Some IoT devices and services may be shared and could have different ownership, policy, and connectivity domains. These challenges need to be addressed in order to build a secure and resilient IoT infrastructure, where Confidentiality, Integrity, and Availability must be assured [8].

D. Intrusion Detection System (IDS) and threat model

Current cybersecurity solutions are not effective to stop the exponential growth in number and complexity of cyberattacks [9]. There are two basic intrusion detection techniques: signature based and anomaly based IDS [10]. Signature based IDS builds a database of known attack signatures. However, these systems cannot detect new types of attacks. The main feature of the anomaly detection approach is its capability to detect new attacks. It defines a baseline model for normal behavior of the system through off-line training, and it considers any activity that deviates from the normal model as anomaly [11]. Improving security and reducing risks in information systems depend heavily on analyzing threats, risks and vulnerabilities in order to identify the appropriate countermeasures to mitigate their exploitations [12]. A general IoT threat model helps to identify threat scenarios and their associated risks [13][14]. When created in the design phase, a threat model can assist in developing the required changes to the design to mitigate potential threats [12][15]. In general the steps to create a general threat model are: 1) Identify attackers, assets, threats and other components, 2) Rank the threats, 3) Choose mitigation strategies, and 4) Build mitigation solutions based on the strategies. In Section IV, we show how to integrate the threat model in the IoT Security Development Framework.

III. IOT ARCHITECTURE FOR SMART CAR SERVICES

IoT services for smart cars can be developed using a hierarchical architecture as shown in Figure 2. The framework consists of four layers: IoT end Devices, Communications, Services, and End Users/Applications.

In the first layer from the bottom (end devices) the device information passes through physical devices to provide the required status information to higher layers and also connect the requests from upper layers to the appropriate physical devices in the car [16][17]. The key components in this layer are the sensors for representing the current state of IoT physical devices (e.g., tire pressure sensor), and the actuators to modify the physical environment to a desired state. The second layer (communications layer) is responsible for providing the required connectivity and communications among all the sensors and actuators associated with IoT smart car services or applications [17][18]. The communication technologies that can be used include Internet, satellite, mobile cellular networks, wireless sensor networks, and internal car network infrastructures (e.g. CAN, I2C, MOST). The Services layer provides common middleware and functions to build sophisticated services in the applications layer. This layer plays an interface role between the application layer in the top level and the communication layer in the lower level [18]. The applications layer provide customized applications or services according to the user needs [17][18]. The access to the IoT through this layer can be through a wide range of devices or applications that can be inside the smart car or geographically dispersed.

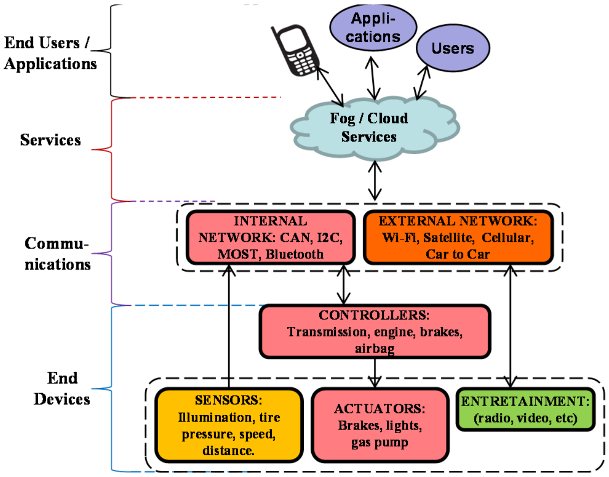
End Users / Applications
Services
Commu-nications
End Devices
Appli-cations
Users
Fog / Cloud Services
INTERNAL NETWORK: CAN, I2C, MOST, Bluetooth
EXTERNAL NETWORK: Wi-Fi, Satellite, Cellular, Car to Car
CONTROLLERS: Transmission, engine, brakes, airbag
SENSORS: Illumination, tire pressure, speed, distance.
ACTUATORS: Brakes, lights, gas pump
ENTRETAINMENT: (radio, video, etc)

Fig. 2. IoT framework for Smart Car

IV. IoT Security Development Framework (ISDF)

The main goal of the ISDF is to provide the architectural support to develop highly secure and trustworthy IoT services that can proactively detect and tolerate malicious behaviors that can be due to attacks, faults (malicious or natural) or accidents.

A. *Secure and Trustworthy IoT Services*

In order to protect against all kinds of attacks, a great variety of security technologies and mechanisms can be used. However, many of these technologies and mechanisms are configured manually and lack support for automated configurations. We define a trustworthy service to be the one that can secure and protect the system against cyberattack (Self-protect), that can continue to operate normally by meeting its performance requirements in spite of faults and destructive attacks (self-healing, and self-optimizing), and can update its configuration and security policies to maintain security, privacy, resilience to faults and accidents, and quality of service requirements (self-configuration). The IoT security that needs to be maintained at all layers can be defined using: 1) Authentication to define the means by which principals securely identify themselves to a system; 2) Authorization for limiting access by principals based on their identity to valued resources; 3) Integrity to detect whether a message or object was modified in an illegal manner; 4) Non-repudiation to prove that certain principals have sent/received a particular message; and 5) Auditing that involves logging security-critical operations that occur within a security domain.

B. *ISDF Architecture*

The ISDF integrates the development of IoT services with security mechanism at the design and development stage as shown in Figure 3. The ISDF is organized as 2-D architecture with four layers and each layer is implemented into four planes: Function Specification, Attack Surface, Target, and Mitigation planes. For each layer, we first identify the Attack Surface (AS) that characterizes the entry points that can be exploited by attackers to inject malicious events or behaviors in the smart car environment, followed by identifying potential objects or functions that can be targeted by attacks, and finally we identify the mitigation mechanisms that can be implemented to mitigate these attacks.

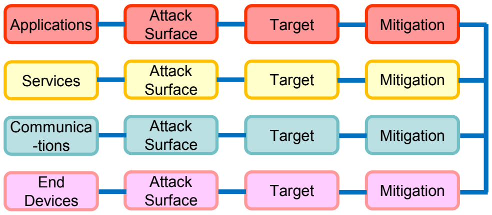

Fig. 3. ISDF main functions.

1) *End-Devices Layer*

Table 1 shows the attack surface, target, impact, and mitigation mechanisms associated with this layer.

TABLE I. End Nodes Layer

Attack Surface	Target	Impact	Mitigation mechanism
Controllers	Control, information	Control, human life safety, time	IDS, behavior analysis
Sensors	Information, access to the system	Control, human life safety, money, energy	Lightweight encryption, IDS, behavior analysis, sensor authentication
Actuators	Control	Control, human life safety, time, money	Lightweight encryption, IDS, ABA, anti-jamming
Entertainment	Access to the system	Time, energy, money	Encryption, moving target defense, ABA

2) *Communications Layer*

Table 2 shows the attack surface, target, impact and mitigation mechanisms associated with this layer.

TABLE II. Communications Plane

Attack surface	Target	Impact	Mitigation mechanism
Protocols	Access, information, control	Control, human life safety, time, money, energy	Authentication, access control, IDS, ABA, anti-jamming
Firewalls	Access to the system	Time, money, providers reputation	IDS, behavior analysis, authentication
Routers	Access, information	Control, human life safety, time	IDS, ABA, anti-jamming
Communication Bus	Information, control,	Privacy, money, human life safety, time	Encryption, IDS, moving target defense, ABA

3) *Services Layer*

Table 3 shows the attack surface, target, impact and mitigation mechanisms associated with this layer.

TABLE III. SERVICES LAYER

Attack Surface	Target	Impact	Mitigation mechanism
Cloud storage	Personal and confidential information	Information, money, time, safety	Encryption, IDS, moving target defense, behavior analysis, selective disclosure, data distortion, big data analysis
Web services	Control, monitor	Control, human life safety, money, information	Authentication, IDS, behavior analysis

4) *Applications Layer*

Table 4 shows the attack surface, target, impact and mitigation mechanisms associated with this layer

TABLE IV. APPLICATIONS LAYER

Attack surface	Target	Impact	Mitigation mechanism
Mobile devices (cellphone, tablets)	Information, control	Human life safety, personal information, money	Authentication, access control, IDS, behavior analysis
Programs / Applications	Access to the system, control, information	Time, money, safety, reputation	IDS, behavior analysis, authentication

V. VEHICLE INFORMATION AND MANAGEMENT PORTAL (VIMP) SERVICES

The VIMP will provide 24/7 visibility into the smart car operational states (e.g., engine conditions, entertainments, etc.) and also provide the mechanisms to secure and protect the smart car operations and functions. Figure 4 shows the main components of the VIMP fields that can be maintained at the portal and the deployment approach for VIMP services.

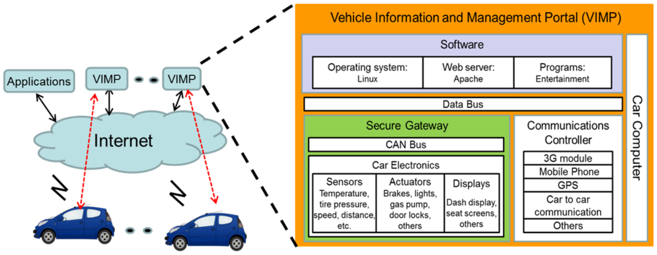
Applications
VIMP
VIMP
Internet
Vehicle Information and Management Portal (VIMP)
Software
Operating system: Linux
Web server: Apache
Programs: Entertainment
Data Bus
Secure Gateway
CAN Bus
Car Electronics
Sensors Temperature, tire pressure, speed, distance, etc.
Actuators Brakes, lights, gas pump, door locks, others
Displays Dash display, seat screens, others
Communications Controller
3G module
Mobile Phone
GPS
Car to car communication
Others
Car Computer

Fig. 4. VIMP fields to be continuously monitored and analyzed.

The ISDF can be used to detect and mitigate any abnormal behavior in the car by using the VIMP. For example, The VIMP can be used to monitor 24/7 an elderly driver and make sure he/she driving is safe. Auto can benefit from getting information about the engine performance. An ABA of the VIMP can detect an ongoing attack against the smart car services and that will trigger a mitigation action to stop the attack or minimize its impact. These are just samples of the many services that can be developed by leveraging the VIMP capabilities. In the next section, we show how we can secure and protect the VIMP services.

VI. SMART VEHICLE TESTBED AND ABA-IDS METHODOLOGY

The SV testbed has the characteristics and functionalities of the actual smart vehicles such as sensors, actuators, automation systems, and communication channels. In our testbed, the user can monitor SV variables and control SV components using a variety of protocols (e.g. Bluetooth, Wi-Fi, and I2C) as shown in Figure 5. Monitored variables include temperature, distance, motion, and illumination. The controlled components are SV lights, movement guidance (left, right forward, backward), brake and speed. The monitoring and control tasks can be performed locally by accessing our secure gateway, or remotely by using the VIMP control services.

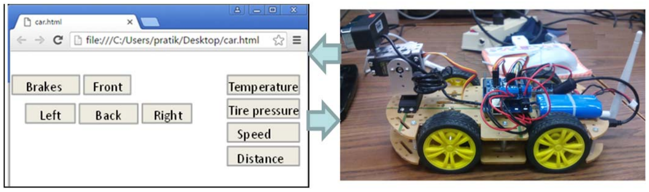

Fig. 5. Smart Vehicle Testbed and VIMP

The VIMP is implemented as a web server, which provides the required communication interfaces to enable all devices (e.g. Cellphone, tablet, and Laptop) to interact and exchange information with the SV. The VIMP server is based on the Raspberry Pi system with Linux. The VIMP provides information about the status of the car, for example distance from objects, temperature of the car, velocity, etc. It also enables users to control some functions such as moving the vehicle in any direction. Attackers may exploit any existing vulnerability to gain access to the system and launch an attack [20]. For instance, the ISDF target at the End Devices layer can be the vehicle's temperature sensor that can be remotely accessed to illegally obtain information. Since sensors usually have low (or no) computational power, it is unrealistic to apply encryption techniques, a more suitable approach is to use ABA. We have developed mechanisms to protect the operations of sensors against any type of threat by ABA of the end devices operations. The main modules to implement our approach are (see Figure 6): 1) Continuous Monitoring, 2) Data Structure, 3) Anomaly Behavior Analysis, 4) Sensor Classification, and 5) Recovery Actions.

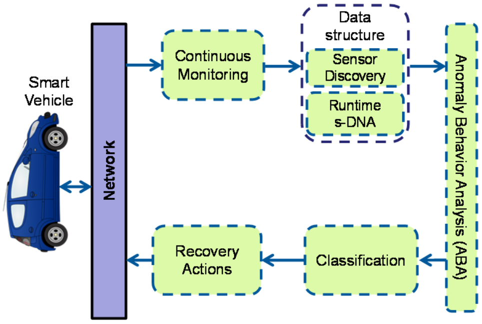

Fig. 6. Methodology for ABA-IDS for sensors

A. *Continuous monitoring*

We have developed software tools to capture the behavior of sensors that are important to characterize their normal operations: source, destination, and content of packets. The sensor's data is extracted from the protocol and sent to the Data Structure module, where the sensor is automatically identified and its runtime profile, that we refer to as s-DNA, is built by using discrete wavelet transform (DWT) coefficients.

B. *Data Structure*

This module performs two tasks: 1) Compute the DWT coefficients form the signal, 2) Identify sensor type based on received data and create the runtime profile to be compared with the reference s-DNA in the ABA module.

1) *DWT coefficients*

The data obtained from the sensor, is decomposed using DWT as shown in Equation (1) and (2).

$$y_{high}[k] = \sum_{n} x[n] * g[2k - n] \tag{1}$$

$$y_{low}[k] = \sum_{n} x[n] * h[2k - n] \tag{2}$$

The original signal *x[n]* is decomposed into an approximation coefficient $y_{high}[k]$, and a detail coefficient $y_{low}[k]$ by applying a high pass *g[n]* and a low pass *h[n]* filters respectively. We use Haar wavelet as the function to extract the coefficients because any continuous function can be approximated uniformly with this function [21]. Once the signal is decomposed, the coefficients are aggregated in a single vector that is used to build the s-DNA data structure.

2) *Sensor classification*

The next step is to identify the sensor type based on the data received. For this task, the Euclidean distance D_j in Equation (3) is computed between the runtime coefficient vector $\boldsymbol{v}$ and a matrix $\boldsymbol{M}$ of coefficients obtained in the offline training phase.

$$D_j = \sqrt{\sum_{i=1}^{n} \left(M_{i,j} - v_i \right)^2} \tag{3}$$

Once the sensor type has been identified, the rest of the data is compared with the coefficients in the same column (*j*) of the matrix to obtain the Euclidean Distance.

C. *Abnormal Behavior Analysis (ABA)*

The Euclidean distance is compared with the reference model in the ABA. The reference model is built during the offline training by using normal measurement attributes. Five vectors are used to find the control limits for normal operation [22]. Once all the distances are computed, the upper and lower control limits (UCL and LCL, respectively) for the normal behavior are calculated using Equations (4) and (5), where $\bar{x}$ is the mean value, σ the standard deviation and α is a sensibility level. For normal control limits α=3 [22].

$$UCL = \bar{x} + \alpha\sigma \tag{4}$$

$$LCL = \bar{x} - \alpha\sigma \tag{5}$$

D. *Classification*

Once the ABA module has determined that there is an abnormality in the data provided by the sensor, the classification unit identifies the type of the observed abnormality. For this task the Euclidean distance is used to detect behaviors and trends. For example in a DoS attack, the distance shows sudden changes above the UCL.

E. *Recovery actions*

When an abnormal behavior is detected, several recovery actions can be taken (e.g. discard data, authenticate the sensor, change network configuration, etc.). However, there is a possibility that the attack cannot be classified (e.g. new attacks), in such cases the data is rejected.

VII. Experiments and Results

A. *Offline Training Phase*

The first step in this phase is to build the matrix for sensor's classification. This is part of the reference model since the runtime data will be compared with the elements of the matrix. Table 5 summarizes the mean value and the limits of normal operation for the sensors used in our evaluation.

TABLE V. Normal Operation Limits

Sensor	Mean	UCL	LCL
Temperature	7.80	11.2	4.3
Motion	3161.08	5238.43	1083.73
Distance	14.54	18.20	10.78
Illumination	6.60	8.67	4.53

Once the system has been trained, the next step is to launch attacks against that sensor to learn its behavior under attacks. The classification of the attacks is based on the Euclidean distance (see Figure 7). A window of seven continuous distances is used to verify any behavior trend [21]. However, for some attacks (e.g. DoS), the seven samples are not needed since the Euclidean distance goes out of the limits.

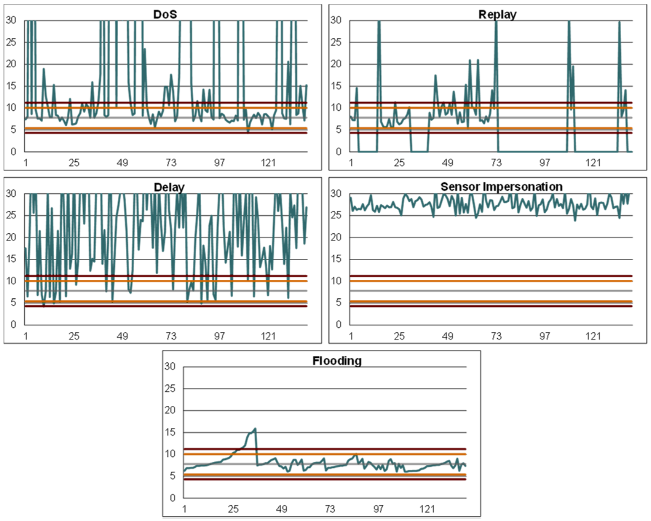

Fig. 7. Behavior for trained attacks.

B. *Online Testing phase*

Once the s-DNA profile is obtained during the offline training phase, the system is tested and evaluated. Notice that some values are above the UCL for the abnormal behavior while other values are in the normal behavior area. This happens because the attacks were performed several times during the experiment. We have evaluated the attacks shown in Table 6 for all the sensors available in our testbed. It also summarizes the detection and classification accuracy of our approach for each type of attack.

TABLE VI. TESTED ATTACKS

Attack	Detection Rate	Classification Rate
Replay Attack	93 %	98 %
Delay Attack	96 %	88 %
DoS Attack	100 %	98 %
Flooding Attack	95 %	98 %
Sensor Impersonation	98 %	85 %
Pulse DoS	100 %	90 %
Noise injection	95 %	95%

From Table 6, pulse DoS and noise injection are two new attacks that were not considered in the offline phase. Here the system detects these attacks and classify them as “new attack”. Two cases that triggered false positives: 1) when the behavior is not considered in the training phase (e.g. environmental manipulation); and 2) When the sensor needs to reach its steady state after an attack. Our experiments show that at most 4.2% of these situations produced false positives alerts. Once the attack is detected and classified, the recovery actions to solve the problem are taken. The actions include: 1) reject sensor’s data, 2) launch an alert, and 3) deauthenticate the sensor. All the data contained in the sample is discarded and the sensor is asked to send data again so that it can be re-authenticated (sensor discovery).

VIII. CONCLUSIONS AND FUTURE WORK

In this paper, we presented an IoT security development framework and show how to apply it to develop secure and trustworthy smart car services. In ISDF, we integrated the threat model with all IoT layers so security and protection mechanisms can be addressed in an integrated manner while we are developing the functions and services of each layer. We also show how to develop Vehicle Information and Management Portal (VIMP) services that provide visibility into the current state of all the components of a smart car and also control and management services. In our experimental results and evaluations, we show how to apply our ABA methodology to secure and protect the VIMP sensor information and operations. Our anomaly behavior analysis methodology includes the use of a sensor-DNA profile (s-DNA) that is developed to accurately characterize normal sensor operations. We have shown that our approach can detect both known and unknown attacks with high detection rates and low false positive alarms (around 4.2%). We have also shown that our attack classification method achieved 98% accuracy for known attacks and up to 95% for unknown attacks (classified as “new attacks”). We are currently extending our approach to secure and protect all other layer functions and services.

ACKNOWLEDGMENT

This work is partly supported by the Air Force Office of Scientific Research (AFOSR) Dynamic Data-Driven Application Systems (DDDAS) award number FA95550-12-1-0241; National Science Foundation research projects NSF IIP-0758579, NCS-0855087, and IIP-1127873; Thomson Reuters in the framework of the Partner University Fund (PUF) project (PUF is a program of the French Embassy in the United States and the FACE Foundation and is supported by American donors and the French government); National Instruments though its NI Research Academic Grant 2015.

REFERENCES

[1] Andrea Z., Nicola B., Angelo C., Lorenzo V., and Michele Z., “Internet of Things for Smart Cities”, IEEE Internet of Things journal, vol. 1, no. 1, February 2014.

[2] S. Chakraborty, M. A. Al Faruque, W. Chang, D. Woswami, M. Wolf, Q. Zhu, “Automotive Cyber–Physical Systems: A Tutorial Introduction”, IEEE CASS, 2016.

[3] D. Kushner, “The Real Story of Stuxnet, How Kaspersky Lab tracked down the malware that stymied Iran’s nuclear-fuel enrichment program”, IEEE Spectrum, February 2013.

[4] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno, “Comprehensive experimental analyses of automotive attack surfaces” USENIX Conference on Security, 2011.

[5] A. Muhammad, D. Ayavoo, M. J. Pont, “A Novel Shared-Clock Scheduling Protocol for Fault-Confinement in CAN-based Distributed Systems”, IEEE 5th International Conference on System of Systems Engineering, 2015.

[6] Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are. (April 2015). [Online] Available: http://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf

[7] H. Suo, J. Wan, C. Zou, J. Liu, “Security in the Internet of Things: A Review”, International Conference on Computer Science and Electronics Engineering (ICCSEE), 2012, vol. 3.

[8] S. Greengard, “Cybersecurity Gets Smart”, Communications of ACM, May 2016, Vol. 58, No. 5, pp. 29-31

[9] O. Can, O. K. Sahingoz, “A survey of intrusion detection systems in wireless sensor networks”, 6th IEEE International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO), 2015.

[10] S. Fayssal, S. Hariri, Y. Al-Nashif, “Anomaly-Based Behavior Analysis of Wireless Network Security”, Fourth Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services, 2007.

[11] Scott Musman, Mike Tanner, Aaron Temin, Evan Elsaesser and Lewis Loren, “Computing the Impact of Cyber Attacks on Complex Missions”, 2011 IEEE International Conference on Systems (SysCon).

[12] D. Xu, M. Tu, M. Sanford, L. Thomas, D. Woodraska, and W. Xu, “Automated Security Test Generation with Formal Threat Models”, IEEE transactions on dependable and secure computing, vol. 9, no. 4, July/August 2012.

[13] R. Schlegel, S. Obermeier, J. Schneider, “Structured System Threat Modeling and Mitigation Analysis for Industrial Automation Systems”, IEEE 13th International Conference on Industrial Informatics (INDIN), July 2015.

[14] N. Pascal, Y. Badr, K. Barbar, F. Biennier, E.K. Lee, P. Chu, R. Gadh “Risk management and security in service-based architectures”, International Conference on Advances in Computational Tools for Engineering Applications, 2009. ACTEA 2009.

[15] Jiong Jin, Jayavardhana Gubbi, Slaven Marusic, and Marimuthu Palaniswami, “An Information Framework for Creating a Smart City Through Internet of Things”

[16] Hiro Gabriel Cerqueira Ferreira, Edna Dias Canedo, Rafael Timóteo de Sousa Junior, “IoT Architecture to Enable Intercommunication Through REST API and UPnP Using IP, ZigBee and Arduino”

[17] Moataz Soliman, Tobi Abiodun, Tarek Hamouda, Jiehan Zhou1, Chung-Horng Lung, “Smart Home: Integrating Internet of Things with Web Services and Cloud Computing”

[18] P.K. Manadhata, J. M. Wing, “An Attack Surface Metric”, IEEE Transactions on Software Engineering, vol. 37, no. 3, May/June 2011.

[19] Mahmud Hossain, Maziar Fotouhi, and Ragib Hasan, “Towards an Analysis of Security Issues, Challenges, and Open Problems in the Internet of Things”, 2015 IEEE World Congress on Services.

[20] S. Mallat, “A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way”, 3rd Edition, Elsevier, ISBN-13: 978-0123743701

[21] D. C. Montgomery, “Statistical Quality Control”, 7th Edition, John Willey & sons, ISBN-10: 111814681.