Anomaly detection in smart city parking data: A case study

Victor Garcia-Font
Internet Interdisciplinary Institute (IN3)
IT, Multimedia and
Telecommunications Department
Universitat Oberta de Catalunya (UOC)
Email: vgarciafo@uoc.edu

Carles Garrigues
Internet Interdisciplinary Institute (IN3)
IT, Multimedia and
Telecommunications Department
Universitat Oberta de Catalunya (UOC)
Email: cgarrigueso@uoc.edu

Helena Rifà-Pous
Internet Interdisciplinary Institute (IN3)
IT, Multimedia and
Telecommunications Department
Universitat Oberta de Catalunya (UOC)
Email: hrifa@uoc.edu

***Abstract*—Wireless Sensor Networks (WSNs) are a principal technology in many smart city projects to collect data from the streets and send it to the data centers of the municipalities. Nonetheless, WSN are easy to attack and, therefore, smart city administrators have to verify that the WSN data is faultless. In this article we present a case study on anomaly detection in smart city parking data using WSN information from a Barcelona smart city provider. In order to analyze this data, we used One-class Support Vector Machines (OC-SVM) in a dataset with application information from the parking sensor readings and also system status data. The results of the anomaly analysis were satisfactory, achieving a 97.13% detection rate and 13.13% false positive rate.**

***Keywords*—Information security, Intrusion detection, Smart cities, Support vector machines, Wireless Sensor Networks**

I. Introduction

In the last few years, cities around the world are building new smart city systems, which rely on advanced communication protocols and the latest technology to improve their operational structure and to acquire a data-driven management perspective. Thus, smart cities gather urban information from a broad range of sources of the Internet of Things (IoT), like social networks, mobile phones and other elements of the smart city. This involves deploying equipment such as wireless sensor networks (WSNs) in the city, a new context where this technology is only partially ripe.

Recently, there has been a major and fast increase in the number of WSN deployments, which has involved many different providers, technologies, solutions, requirements, etc. As a consequence of this, security has been put aside in many implementations. In most of the cases, the public administration has outsourced the WSN implementations to external providers, who are responsible for the network administration. Thus, smart city managers are excluded from the network security administration and they cannot even verify that the WSNs are operating without attacks or failures. The information that WSNs supply is essential for planning, operation and evaluation tasks of several municipality departments, private companies, facilities, etc. Therefore, public administrations need to be confident that their WSNs are being operated faultlessly and the data gathered by the sensors is reliable. In this scenario, the first step towards improving security is recovering the lost visibility over the networks operated by third parties.

We have analyzed a specific scenario from the city of Barcelona: the parking service data. Nowadays, Barcelona is taking a leadership role in the smart city context and is developing *CityOS*[1], an operating system for cities that aggregates modules of data processing, analysis, historical data management, BI, etc. A major objective of the smart city of Barcelona is to deploy a system for the easy integration of third party modules. For example, *CityOS* includes the module *City Service Development Kit* (*CitySDK*)[2], which offers a set of open source tools to facilitate cities to open their data and to help developers to create digital services for the city. Other remarkable projects included in the smart city of Barcelona are *Sentilo*[3], *iCity*[4] and *Open Cities*[5].

Normally, smart city systems are designed as service oriented architectures, which can be conceptually divided in three layers[1]. The first layer includes the elements that collect information from the city (e.g. sensors, actuators, surveillance cameras, social networks, SCADA). The second layer acts as a middleware, which provides the city with connectivity APIs and multiple services such as cloud computing, analytics and storage. Finally, the third layer is an application layer, in which the city council and third parties implement applications based on the data and the services offered by the middle layer. In general, these architectures aim to maximize interoperability among modules in order to encourage development of third party applications and to ease access to the services and city data.

In this case study, we worked on improving second layer smart city algorithms in order to increase the visibility of the first layer incidents in the specific scenario of a parking service. We analyzed parking data gathered by WSNs with algorithms and resources available at the middle layer.

The rest of this paper is structured as follows: Section II contains related work and background. The data analysis of

[1]"CityOS", http://ibarcelona.bcn.cat/ca/o-government/city-os
[2]"CitySDK", http://www.citysdk.eu
[3]"Sentilo", http://www.sentilo.io
[4]"iCity", http://www.icityproject.com
[5]"Open Cities", http://opencities.net

this case study is explained in Section III. Finally, Section IV concludes the paper.

II. Related work

In order not to exclusively depend on prevention countermeasures, Intrusion Detection Systems (IDSs) are deployed as additional defensive barriers to alert administrators when unusual situations are taking place in the system. In this section, we briefly review basic intrusion detection techniques and we put the focus on anomaly detection based on Support Vector Machines (SVM). As we show in this section, SVMs have several properties which make them suitable for the heterogeneous context of this case study.

A. Intrusion Detection

Principally, detection techniques are based on misuses or anomalies. Techniques based on misuses rely on an extensive database of attack signatures. An attack signature is a pattern that can be used to identify an attacker's attempt to exploit a known operating system or application vulnerability. Alarms are raised when new observations match any of the signatures[2]. The main advantage of this type of detection is the low rate of false-positives. Nevertheless, it has the drawback of not being able to detect unknown threats for which there is not yet any implemented signature.

Anomaly-based techniques are based on identifying differences between the actual network activity and a predefined model considered as normal. Different techniques are used to define what to consider normal[3]. The most widespread techniques come from the Machine Learning (ML) domain. Unlike misuse-based detection, anomaly-based techniques are suitable to detect unknown attacks. On the other hand, as main drawbacks, these techniques trigger high false-alarm rates and the normal-activity profiles need to be periodically updated in networks with dynamic activity. In the next section, we describe a popular ML technique to discover anomalies based on SVMs.

B. Support Vector Machines

Classification techniques based on SVMs have proven to be effective in several contexts related with intrusion and anomaly detection[4], [5]. Basically, classification techniques based on ML require two steps. First, a dataset is used to train a model. Then, the trained model is used to classify new data samples. Several features define each sample of the datasets. The SVM classification process represents the training dataset in a n-dimensional vector space, n being the number of features of the training data. Then, it defines a hyperplane (i.e. a $n-1$ dimensional plane) that separates with a maximum margin the samples from the different classes. The Support Vectors (SV) are the subset of training samples that are near the hyperplane and that are defining it. Finally, the hyperplane acts as a frontier to classify other samples. Thus, SVs are capable of representing complex models with few samples from the training dataset.

In ML the classification techniques are divided in three types: supervised, semi-supervised and unsupervised. Supervised ML techniques require to label each sample of the training dataset with the class that it belongs. Unsupervised ML techniques do not require labels. Finally, in the scope of anomaly detection using a binary class with the labels normal and abnormal, semi-supervised ML techniques only require labeled normal data. In the scope of anomaly detection in a smart city, supervised ML techniques are not adequate, because it is challenging to obtain labeled data (particularly for anomalous samples).

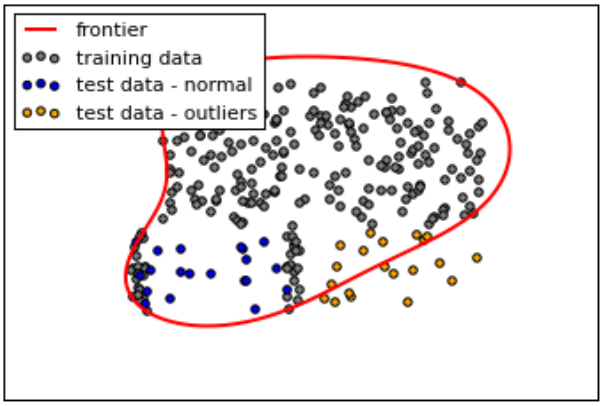
Consequently, in this case study we use One-class Support Vector Machines (OC-SVM), which are a special case of unsupervised SVMs that do not require labeled data. OC-SVMs build a frontier to classify new samples as normal or outlier. In order to build this frontier, Radial Basis Function (RBF) kernel OC-SVMs use basically two parameters[6]. On the one hand, ν defines the maximum fraction of outliers present in the training data. On the other hand, γ establishes the influence area of the SVs on the classification. Figure 1 exemplifies the impact on the learned frontier that γ has for a fixed value of ν. As can be seen in Figure 1b, increasing the value of γ implies to adjust the frontier closer to the training samples. This reduces the number of misclassified outliers as normal samples. However, Figure 1c shows that increasing γ too much causes overfitting the training data. A reliable approach to select optimum parameters is grid search[7]. In this method a grid with parameter values is exhaustively explored in order to select the values that give the best performance of the SVM over a set of samples.

III. Anomaly detection in a parking WSN

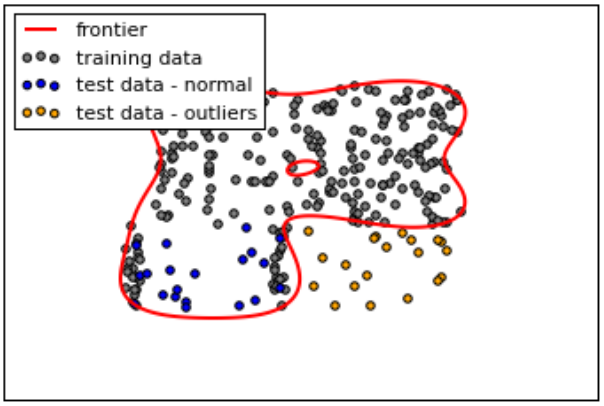
As discussed earlier, smart cities gather a vast amount of WSN data, which must be analyzed in order to verify that it does not contain errors. These errors might be accidental or, as we analyze in this article, the result of a malicious attack. The purpose of this case study was to take the first step on this verification process setting up the focus on WSN parking data.

Parking service is nowadays one of the most characteristic services in the present smart cities. In order to implement a parking service, providers usually deploy hundreds of sensors on the streets grouped in WSNs. The sensors detect the presence of a car in a parking slot and send this information cooperatively using lightweight protocols (e.g. 802.15.4, ZigBee, 6LoWPAN) to a gateway. The gateway has a long distance connection with the smart city data centers (e.g. xDSL, 3G). Finally, the information at the data center is supplied to several city services such as cell phone applications in order to inform the citizens about occupancy in the different areas of the city.

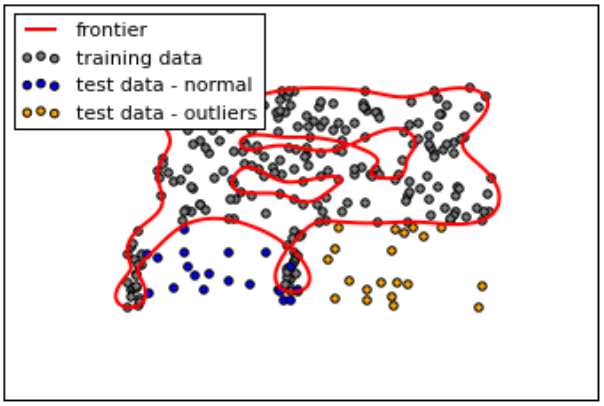
The following sections describe an anomaly analysis on WSN parking data. First, we generated datasets to train and test OC-SVMs (Section III-A). Second, we defined several feature vectors to be compared (Section III-B). Finally, we carried out the analysis and we discuss the results (Section III-C).



(a) $\nu = 0.01$, $\gamma = 0.2$



(b) $\nu = 0.01$, $\gamma = 0.8$



(c) $\nu = 0.01$, $\gamma = 2$

Fig. 1. Radial Basis Function (RBF) kernel OC-SVMs trained with different parameters. In a, the influence area of the SVs is wide, so many outliers are incorrectly classified as normal. In b, the frontier is very near the SVs, so there is a reduced number of misclassified outliers and all the normal test samples are properly classified. In c, the model is overfitting the training data, so many normal test samples are classified as outliers.

A. Dataset generation

In order to create the parking datasets for this case study, we started out with a dataset from a parking sensor provider of Barcelona. This dataset could be considered as anomaly-free. Then, we generated a wider dataset with approximately the same behavior as the original data. With the goal of building more comprehensive scenarios, the parking data was supplemented with WSN simulations made with Castalia[6]. The extended dataset was used to train the ML models.

We also generated five test datasets, which contain two datasets without anomalies and three datasets with anomalies due to attacks affecting the entire duration of each sample. The two anomaly-free datasets were generated using two different WSN configurations: a one-hop and a multihop. The three datasets with anomalies were created including the most representative attacks for the basic WSN layers using the same types of WSN configuration:

- **Jamming.** This attack at the physical layer consists in sending a high power signal to the gateway in order to avoid a correct reception of legitimate packets. This attack was performed against the one-hop configuration.
- **Exhaustion.** At the data link layer, the attackers exploit the channel access protocols to prevent transmissions from legitimate nodes. The attackers continuously occupy the communication channel impeding other transmissions. This attack was performed against the one-hop configuration.
- **Selective forwarding.** Attack at the network layer. The attackers have captured a node that stops re-transmitting packets from some nodes. This attack was performed against the multihop configuration.

We assumed that attackers intended to gain advantage over other users of the public car park. To this end, the attackers disrupted the communication of several nodes during high occupancy hours. Thus, parking applications did not receive updates when parking slots became free and, therefore, attackers had a higher probability to find available slots in certain areas. We considered that two hours was the minimum amount of time that the attackers would need to disrupt the communication to gain a real advantage over the other drivers. Thus, simulations had an extension of two hours.

The dataset with the parking readings included the following fields: sensor identifier, timestamp, gateway identifier, sequence number and current state of the parking slot (free/occupied). This original dataset was used to prepare the Castalia simulations with the WSNs configured in one-hop and multihop. The simulation outcomes were used as the final training and test datasets with the following main variables: **application layer data**, which includes the number of changes in the parking slot and the number of lost application packets, and **system status data**, which includes the number of received radio packets with and without interferences, ACKs, CTSs and RTSs.

B. Feature vector definition

In order to detect anomalies in an heterogeneous context with a large number of data sources such as the smart city, we used OC-SVM. In this section, we define different feature

[6]"Castalia 3.3", http://castalia.npc.nicta.com.au

vectors to train and test the OC-SVMs. The feature vectors determine the set of variables included in the models and, therefore, these variables are the basic knowledge to decide if a dataset contains anomalies.

As we have seen in Section I, smart city equipment and, consequently, their system status data is not always available to the administrators. Therefore, it is necessary to base the anomaly analysis on application layer data. However, application layer data is often not enough to discover certain failures or attacks. In some situations, it is necessary to supplement this data with some system status data from the accessible devices. However, classification techniques normally achieve lower accuracy when vectors have too many dimensions or when several features are correlated. Therefore, building feature vectors combining application and system status data could decrease the performance results of the OC-SVMs. For these reasons, in this case study we built various vectors with different features and we compared the results of the analysis under these scenarios. For the one-hop WSN configuration, we defined an scenario without access to any WSN device (feature vector 1) and also another scenario where smart city administrators were able to access system status data at the gateway (feature vector 2). We also tested whether the combination of all the available data decreases the detection results (feature vector 3). Accordingly, we defined different feature vectors containing the following variables:

- **Feature Vector 1**. This included the time of day and, for each sensor, the number of state changes (free/occupied) and the number of lost application packets.
- **Feature Vector 2**. This included the time of day, the number of ACK, CTS and RTS packets sent by the gateway, the number of packets received with and without interferences.
- **Feature Vector 3**. This feature vector included all the previous variables.

In multihop configurations, system status data is normally even more inaccessible and, sometimes, application data is aggregated in intermediary devices, which results in loss of application layer information such as the sequence number of the application packet. Therefore, for this configuration we compared the detection results only using information about the sensor readings (feature vector 1) with also system status information at the application layer about the percentage of lost packets (feature vector 2), which can be calculated with the difference on the sequence number of two consecutive application packets. In this case, we also tested the combination of all these features in a single vector (feature vector 3). Thus, we defined the following feature vectors:

- **Feature Vector 1**. This included the time of day and the number of state changes (free/occupied) detected by each sensor.
- **Feature Vector 2**. This included the time of day and the number of lost application packets.
- **Feature Vector 3**. This feature vector included all the previous variables.

TABLE I
METRICS

Detection rate	$\frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$
False alarm rate	$\frac{\text{false positives}}{\text{false positives} + \text{true negatives}}$
F-score	$\frac{\text{true positives}}{\text{true positives} + (\text{false negatives} + \text{false positives})/2}$

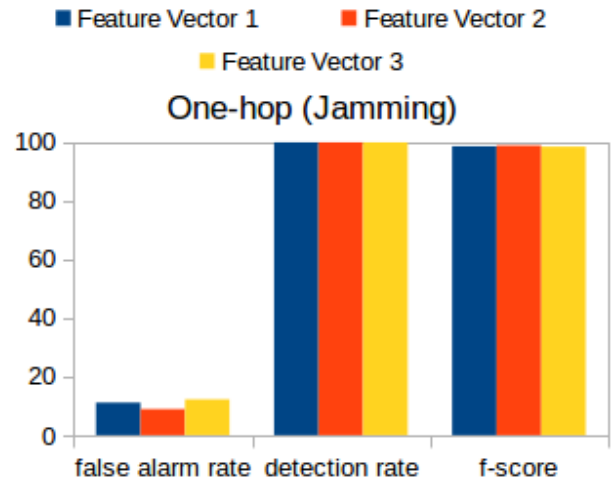
C. Results

In this section, we discuss the results of using OC-SVMs with the previously described feature vectors. In order to evaluate the detection results, we calculated the detection rate, the false alarm rate and the f-score. These are three standard metrics widely used to assess IDSs and ML algorithms [8]. The metrics are detailed in Table I. On the one hand, the detection rate measures the percentage of outliers that have been discovered. On the other hand, the false positive rate indicates the percentage of normal samples misclassified as outliers. Finally, the f-score is a general correctness measure of the test.

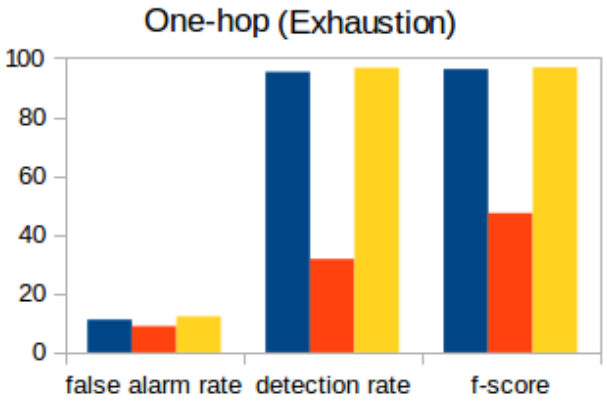
Figure 2 shows the results of the anomaly detection analysis using OC-SVMs trained with $\nu = 0.05$ and $\gamma = 0.001$. Grid search was used to define the value of these parameters. As the charts show, it is necessary to combine data from the application layer and the system status to obtain a general effective detection rate for the diverse WSN configurations and to detect different types of attacks at the different layers. The detection rate in Figure 2a shows that both application (feature vector 1) and system status (feature vector 2) data were perfectly able to detect the jamming attack at the physical layer. However, Figure 2b shows that only application layer data (feature vector 1) detected the exhaustion attack properly. Figure 2c shows that the sensor readings only (feature vector 1) are not enough to detect the selective forwarding attack. Nonetheless, for all the situations, the feature vectors defined with a combination of all the data performed very well achieving an average detection rate of 97.13%. On the other hand, the false positive rate was on average 13.13%. The false positive rate in Figure 2a and Figure 2b was the result of the analysis of the anomaly-free dataset of the one-hop WSN configuration. In Figure 2c, the false positive rate was calculated with the anomaly-free dataset of the multihop configuration. The OC-SVMs that we used were configured to prioritize a robust system against attacks, which implied a high detection rate, but also caused a slight increase of the false positive rate.

IV. Conclusions

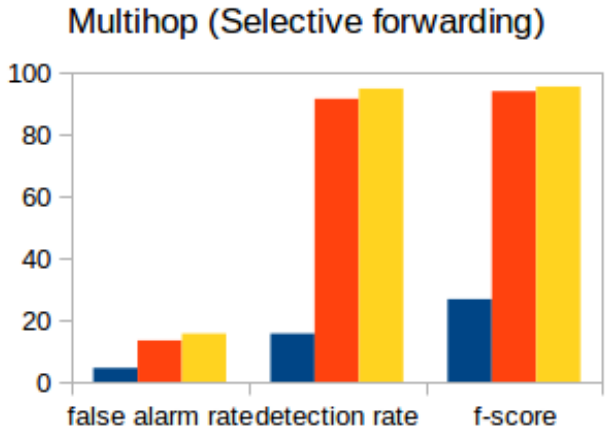
Nowadays, smart city managers are outsourcing the deployment of urban WSNs to external providers. This results in a visibility loss over the WSNs and a global decrease in security. In collaboration with the smart city of Barcelona, we worked on recovering the lost visibility in order to improve



(a) One-hop with jamming attack



(b) One-hop with exhaustion attack



(c) Multihop with selective forwarding attack

Fig. 2. Anomaly detection results for a one-hop WSN under a jamming and an exhaustion attacks, and a multihop WSN under a selective forwarding attack. Three cases with different feature vector definition are compared.

the reliability of the city services. With this goal, we began analyzing WSN parking data to detect anomalies. In this case study, we combined the parking sensor readings with system status data to train and test OC-SVMs to discover anomalies. The test datasets contained anomalies caused by the most representative attacks at the physical, data link and network layers. The performed anomaly analysis achieved a 97.13% detection rate and a 13.13% false positive rate.

V. Acknowledgments

This work is partially funded by the Ministry of Economy and Competitiveness through the projects CO-PRIVACY (TIN2011-27076-C03-02) and SMARTGLACIS (TIN2014-57364-C2-2-R) and for the Government of Catalonia through the subvention of the industrial doctorate ECO/2497/2013 jointly conducted with Cast Info. Barcelona City Council and openTrends deserve particular thanks.

References

[1] V. Garcia-Font, C. Garrigues, and H. Rifà-Pous, "Seguridad en smart cities e infraestructuras críticas," in *Reunión Española sobre Criptología y Seguridad de la Información (RECSI)*. Universidad de Alicante, 2014, pp. 221–226.

[2] I. Kim, D. Oh, M. K. Yoon, K. Yi, and W. W. Ro, "A distributed signature detection method for detecting intrusions in sensor systems," *Sensors*, vol. 13, no. 4, pp. 3998–4016, 2013.

[3] I. Butun, S. D. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 266–282, 2014.

[4] A. Este, F. Gringoli, and L. Salgarelli, "Support vector machines for tcp traffic classification," *Computer Networks*, vol. 53, no. 14, pp. 2476–2490, 2009.

[5] S. Kaplantzis, A. Shilton, N. Mani, and Y. A. Sekercioglu, "Detecting selective forwarding attacks in wireless sensor networks using support vector machines," in *Int. Conf. Intelligent Sensors, Sensor Networks and Information*. IEEE, 2007, pp. 335–340.

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[7] L. Zhuang and H. Dai, "Parameter optimization of kernel-based one-class classifier on imbalance learning," *Journal of Computers*, vol. 1, no. 7, pp. 32–40, 2006.

[8] A. Lazarevic, L. Ertöz, V. Kumar, A. Ozgur, and J. Srivastava, "A comparative study of anomaly detection schemes in network intrusion detection." in *Int. Conf. Data Mining*. SIAM, 2003, pp. 25–36.