Reputation-Based Security Framework for Internet of Things

Ion Bica(✉), Bogdan-Cosmin Chifor, Ştefan-Ciprian Arseni, and Ioana Matei

Faculty of Information Systems and Cybersecurity,
"Ferdinand I" Military Technical Academy, 050141 Bucharest, Romania
{ion.bica, bogdan.chifor, stefan.arseni,
ioana.matei}@mta.ro

Abstract. Mobile crowdsensing has emerged as a new paradigm in the IoT world, exploiting users' mobility in conjunction with advanced capabilities and proliferation of mobile devices. Smartphones, tablets and smartwatches are now typically equipped with sensing and wireless capabilities, enabling them to produce and upload data for different IoT applications. The mobile crowdsensing approach has the advantage of being cost-effective, while also providing real-time data. However, a number of challenges should be addressed in order for mobile crowdsensing to reach its full potential. Security, privacy and reliability of the data provided by mobile devices are the most important ones. In this paper, we propose a security framework with a multi-layer architecture that addresses the trust evaluation of sensing devices based on reputation scores calculated using a naive Bayes algorithm.

Keywords: Mobile crowdsensing · Security framework · Trust management

1 Introduction

The Internet of Things (IoT) refers to a network of interconnected "smart objects" that have incorporated the technology needed to detect and communicate data about their internal state, as well as interacting with the external environment. One direction of development in IoT is currently represented by mobile crowdsensing. The devices that we carry with us every day (such as smartphones, tablets, smartwatches) are equipped with several physical and virtual sensors that may collect and share information about the surrounding environment for different purposes.

Mobile crowdsourcing has attracted the attention of researchers with applications designed for air quality monitoring [1], traffic monitoring [2] or intelligent parking [3, 4]. The idea behind mobile crowdsensing applications is to reduce costs by replacing or complementing traditional wireless sensor networks. A conventional sensors network in IoT is usually intended for a specific application, but mobile crowdsensing is trying to reuse data for multiple purposes [5]. There are a series of researches regarding the definition of frameworks for mobile crowdsourcing [6, 7], as well as specific implementations [8, 9] that allow the development of applications by reusing the data from multiple sensors.

E. Simion and R. Géraud-Stewart (Eds.): SecITC 2019, LNCS 12001, pp. 213–226, 2020.
https://doi.org/10.1007/978-3-030-41025-4_14

The main drawback of mobile crowdsensing is finding out a method of establishing the degree of trust of the sensing nodes within the network, because they may affect the quality of services provided. In crowdsensing applications, devices involved in the sensing process are vulnerable and they can insert erroneous data into the system either intentionally (attacks of malicious people) or unintentional (environmental disturbances). Consequently, it is challenging to ascertain the correctness of the collected data and is difficult to establish the reliability of it without knowing whether the data is valid or not.

This paper proposes a security framework with a multi-layer architecture that addresses the trust evaluation of sensing devices based on reputation scores calculated using a naive Bayes algorithm. The proposed framework consists of interconnected modules that are integrated at each of the main layers of an IoT system: Cloud, gateway, and device. The framework is built on a customized decentralized architecture, empowering middle-layer devices, such as gateways, while having a central point of management through a Cloud platform. Following the gateway-centric model, our framework moves the main part of the security logic at the gateway layer, where we integrate the core of the reputation-based trust management system.

The framework's key components are presented in the remaining sections of the paper, which has the following structure. Section 2 presents the related work being done in this domain. Section 3 describes the architecture of the proposed framework, followed by Sect. 4, in which the tests and analyses are presented. Section 5 ends the paper with conclusions and future research directions.

2 Related Work

In distributed and collaborative systems, trust management plays a significant role. Ensuring a high degree of trust and security is a critical issue that must be considered when designing a mobile crowdsensing application. Reputation is a concept closely related to establishing a trust relationship between participants. Based on previous experiences and the reference information already collected, a degree of trust or mistrust can be assigned to each participant. Recent studies present an overview of trust management in IoT, explaining its usefulness in a security framework and how it should be exploited. In [10], the security objectives of a trust management system are presented and a review of the current research that deals with the subject of trust in IoT systems is made. It also presents a conceptual model for a holistic framework that contains elements of trust management at each layer and cross-layers. Another detailed study of trust management techniques is described in [11], where a series of frameworks that are based on node reputation are presented: AETS (Adaptive Trust Estimation Scheme), ATBP (Adaption Trust-Based Protocol), TDFDS (Trust-based Development Framework for distributed systems), CTMS-SIOT (Context-based trust management system for the social Internet of Things), etc. The last one is presented in the context of dynamic systems that want to maintain a realistic approach. Regardless of the nature of the architecture (centralized or decentralized), CTMS-SIOT depends on both the past interaction and future prediction and is based on two modules: one for storing contextual trust and one for calculating reputation.

A trust management system based on reputation can defend a network against attacks at nodes level because it facilitates the detection of untrustworthy entities, thus contributing to the decision-making process. Today, there are several proposals and algorithms for computing reputation based on K-Nearest Neighbors, naive Bayes Case-Based Reasoning (CBR) [12] or Fuzzy logic [13]. In [14], the author uses Bayesian inference and self-observation to evaluate trust based on feedback received from neighboring nodes. The proposed model updates the confidence level of the nodes in real-time in order to prevent opportunistic attacks. A different approach to trust calculation is provided in [13] using Fuzzy logic. The system allows the nodes to interact with each other, recording all transactions, then evaluates the performance of each node based on the package delivery ratio (PDR).

A security framework that relies on the trust management module can bring improvements to an IoT architecture in terms of detecting abnormal node behaviors and isolating them. An approach to such a security framework for IoT is presented in [15, 16]. They address the possibility of building services only on the basis of information received from trusted nodes. The information is actually the feedback sent by the neighboring nodes or from the gateway. A slightly different approach is presented in [17, 18] which implements an identity-based key agreement framework to prevent attacks outside the network and to recognize malicious nodes.

To address the problems that appear at all layers in an IoT architecture, we have defined a modularized security framework that allows a decision to be made in accordance with the reliable information collected from the devices that can be used in crowdsensing architectures. Compared to the above-mentioned frameworks, the reputation module is deployed at the gateway layer so that the gateway can select the devices that contribute to data in the mobile crowdsensing architecture.

3 Proposed Architecture Design

The security framework, detailed in the following subsections, makes use of the advantages that reputation-based trust management has, for enforcing the distribution of valid data throughout the system and mitigating different types of attacks. Following the gateway-centric approach that many IoT systems are based upon, we propose a security framework that empowers the gateway as its central element. In this scenario, the Cloud component plays a secondary role, ensuring the communication between the gateway and crowdsensing devices, data consumers or static nodes.

The system architecture contains the following modules: the IoT end-points, the gateway, and the Cloud. The IoT layer comprises devices that produce aggregated data using the on-board sensors and the most trusted crowdsensing information. The gateway layer is the most critical part of our system, being the element that computes the IoT device's reputation and acts as a communication bridge for the local IoT data flow and for uploading the local computed IoT data to an upstream application. The Cloud layer is used to manage local gateways, along with establishing the trust relations between them, and acts as a passive repository for storing the IoT generated data. This architecture is based on a mobile crowdsensing model that enables a collaboratory IoT data delivery application. Thus, the core element of this system is a local network of static IoT devices that generates and aggregates data. These local IoT modules are either low-cost devices or devices which need data generated by other mobile devices located in the environment. The mobile crowdsensing model reduces the cost of the static IoT group deployment by allowing an IoT device with a small number of on-board sensors (simple hardware design) to virtually extend it's sensor capabilities. This mechanism also improves the IoT static group flexibility, by handling other types of sensor data without having to re-deploy the entire sensor fleet. The system architecture is depicted in Fig. 1.

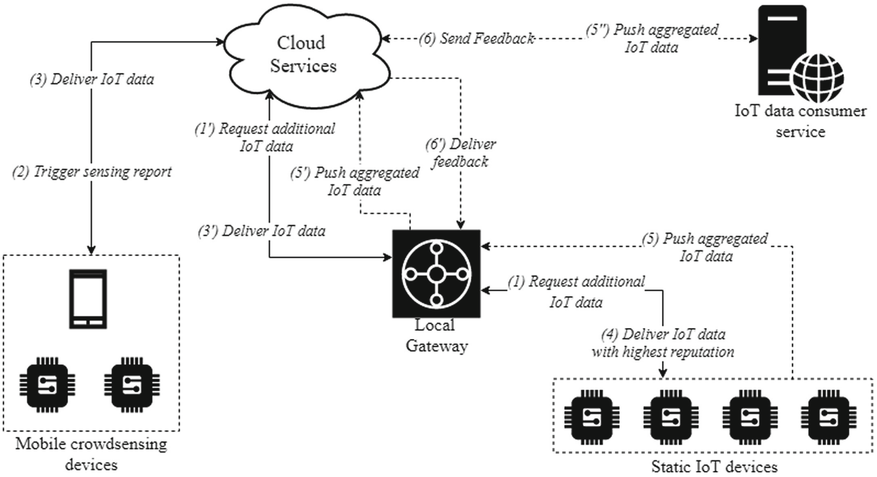
Cloud Services
(6) Send Feedback
(5'') Push aggregated IoT data
IoT data consumer service
(3) Deliver IoT data
(1') Request additional IoT data
(6') Deliver feedback
(2) Trigger sensing report
(5') Push aggregated IoT data
(3') Deliver IoT data
(5) Push aggregated IoT data
(1) Request additional IoT data
Local Gateway
(4) Deliver IoT data with highest reputation
Mobile crowdsensing devices
Static IoT devices

Fig. 1. The architecture of the system

As it can be observed, the static group of IoT devices is extending it's sensor capabilities with the aid of the mobile crowdsensing IoT devices. Thus, when a static IoT device needs additional data, it chooses the information published by the most trusted mobile crowdsensing module (the device with the highest reputation within a certain data category). For instance, if a static device is not equipped with a temperature sensor, it may choose to query the gateway, which in turn delivers the most trusted information provided by a mobile crowdsensing device. By using this approach, the static device can aggregate data from various sources (on-board and participatory sensors), and deliver the information to an upstream application. The upstream application consumes the static IoT delivered data and provides feedback based on the information quality/relevance. If it receives positive feedback, the static device rewards the mobile crowdsourcing module which contributed to the delivered information.

3.1 IoT Device Layer

As mentioned before, the IoT device layer comprises two groups: mobile crowdsensing and the static IoT group which communicate through the Cloud services and the local gateway. The mobile crowdsensing IoT device group is composed of sensors that sample data from the environment and voluntarily submit it to a local gateway. The information submit process is orchestrated through a smartphone application that acquires data from two sources: on-board (local) smartphone sensors and wearables. The smartphone application acts as a data aggregator and submits the information to a local gateway, through the Cloud module, following the mobile crowdsensing paradigm. The controller application acquires sensor data using the following mechanisms:

- it uses the smartphone operating system API's to sample data using the on-board sensors (e.g. use the Android API to query the barometer sensor in order to detect changes in air pressure).
- it uses a low energy connection (e.g. BLE) with wearables in order to extract the sensor data. The controller smartphone application uses the management API exposed by the wearables (e.g. smart-watch, smart-bracelet).

The wearables along with the smartphone onboard sensors share the same trust domain or use an already existing security link (e.g. authentication between the smartphone and the wearable), thus an additional security mechanism is not required. The user device-generated data is relevant only for a certain geographical area, thus the data sampling process is triggered by the smartphone controller, only when the user is located within the local gateway's area of interest. Taking into consideration that the mobile crowdsensing data is consumed based on the reputation value, the controller application generates an identity and uses that identity every time a sensor data is submitted to the gateway. The controller's identity consists of a pair of asymmetric cryptographic keys, each mobile crowdsensing report being signed with the controller's private key. The application controller communicates with the gateway through a data submission protocol, which consists of the following steps:

1. at start-up, the controller application generates an asymmetric key pair and submits to the Cloud service, the public key along with a pseudonym. This tuple represents the application controller's identity.
2. the gateway initiates a report submission session, by sending a request to the Cloud service, which in turn relays the request to all the devices within a geographical area. The session metadata consists of a unique session identifier (randomly generated) and a data category (e.g. temperature, noise).
3. if applicable, the controller application acquires data from the local smartphone and from the connected wearables, aggregates the data in a report, appends the session metadata and signs the report with his private key.

After a member of the mobile crowdsensing IoT group submits a sensing report to the local gateway, the information is stored on the gateway side for a period of time. The mobile device does not have a direct communication link with the gateway, the communication being established by means of the Cloud platform. The mobile device to gateway communication consists of the following steps:

1. the gateway triggers a data sensing query by sending a request to the Cloud platform. The request contains the gateway GPS location, taking into consideration that the mobile crowdsensing data is relevant only for the gateway's proximity.

2. the Cloud platform relays the sensing request to all mobile crowdsensing applications which are located in the gateway's proximity.
3. the targeted mobile application controllers trigger a data sampling process.
4. after the controller mobile application acquires the data, it sends the response to the Cloud platform, which in turn relays it to the gateway.

The Cloud-based communication between the mobile application and the gateway requires only a data connection on the user's smartphone. Although the gateway has communication capabilities (acting as a hotspot or as a base station for the static IoT devices), scanning and subscribing to different networks is a battery intensive task for a smartphone. This is an import factor, taking into consideration that the mobile crowdsensing is not the primary task of a smartphone, and such a solution must be non-intrusive from the performance and user-experience perspectives.

During this time interval, the data is eligible for being consumed by a member of the static IoT group, if the data producer's reputation is the highest within a category. The reputation of the mobile IoT device is computed locally, but it can be transferred from a gateway domain to another, thus the device must use the same identity in order to preserve the reputation value. If a member of the static group needs additional sensor data, it executes a sensor query and the gateway returns the most trusted data within the requested category. After computing the aggregated data with the aid of a mobile device, the static device publishes the information (through the gateway) to a higher layer application that consumes the information. This can be either a smartphone application or a web application that delivers data to end-users or to another IoT device. The gateway exposes an API that allows the data consumer (e.g. end-user smartphone application or web service) to provide feedback for the delivered data. In accordance with the feedback, the gateway increases or decreases the reputation of the participatory sensing device. The transaction is asynchronous because the mobile sensing data can be queried by a static device anytime during the data time-to-live interval, with the gateway acting as a buffer for storing the most recent published information. The gateway publishes the information received from the static IoT devices to the Cloud platform, which in turn relays it to the consumer applications. The data is delivered to the consumer application through a TLS channel, each consumer application having an identity registered on the Cloud platform. The feedback is also delivered to the gateway via the trusted Cloud communication channel, thus the feedback cannot be altered or submitted multiple times.

3.2 Gateway Layer

The gateway module is responsible for computing the reputation of the mobile crowdsensing devices that contribute with sensor data to the static IoT modules. The crowdsensing devices do not share a trust relationship with the gateway, these contributing with information in an ad-hoc manner. By using a reputation algorithm, the gateway delivers to the static IoT device the most trusted information within a category. If a device contributes with relevant information constantly, its' reputation value will be increased, otherwise, the reputation level will decrease if a transaction is considered failed. For computing the reputation level, a naive Bayes algorithm is used. This algorithm was chosen because it does not require high computational resources, being adequate for resource-constrained gateways. In an IoT network, the number of deployed gateways is high, given that these are part of the leaf network segment. Taking this into consideration, low cost gateways are critical in the cost-effectiveness of an IoT application. Thus, a lightweight algorithm like naive Bayes can be executed on general purpose gateways that do not have security as a primary task.

The gateway maintains a repository with the reputation level for each mobile crowdsensing device that submits a sensing report. This repository can be modified only by the naive Bayes algorithm and the reputation value can be transferred to another gateway domain. Taking into consideration that the crowdsensing devices are mobile, there is a low probability for the same device to submit data to the same gateway multiple times, thus the reputation must be transferred from one gateway to another. Given the trust relationship between the gateways, when a new device submits data into a zone, the gateway sends a broadcast request to all gateways in order to find a baseline reputation score. The communication between the gateways is achieved by means of the Cloud platform, which relays the messages. The gateway that executes the query chooses the minimum reputation score received from other gateways and uses this value as the baseline reputation level for the newly registered crowdsensing device.

As stressed before, the naive Bayes method was chosen due to its simplicity, which assumes that an agent can deliver information with the characteristic that one delivered feature is independent of the others. For instance, in our crowdsensing IoT scenario, the naive Bayes paradigm is translated into the characteristic that a mobile user can deliver a trustworthy temperature value without influencing the trustworthiness of the delivered air pressure value. In Fig. 2 is depicted the structure of the proposed naive Bayes network. The purpose of our naive Bayes algorithm is to predict the probability of a mobile device to deliver trustworthy information, based on the previously delivered data.

As presented in Fig. 2, the root node of the naive Bayes network indicates if the mobile agent is trustworthy and the leaves contain the sensor data features. The features are represented by the agent delivered data type (e.g. temperature, CO2) and by meta-information generated by the gateway (e.g. how fast and how often a mobile agent uploads a sensing report).

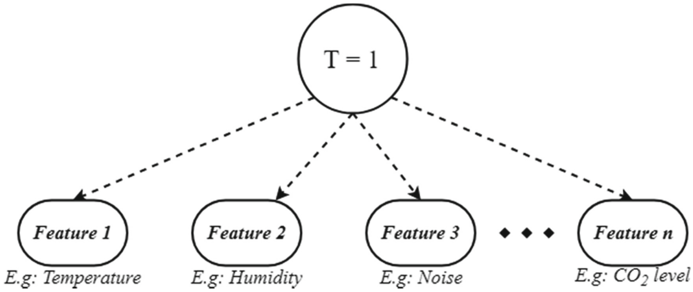

Fig. 2. The naive Bayes network

The local gateway maintains a naive Bayes network for every mobile crowdsensing agent. In order to increase the reputation value, each transaction must be evaluated and classified as satisfying or unsatisfying (Formula 1).

$$p(T=1)=\frac{\#\,of\ successful\ transactions}{\#\,of\ total\ transactions} \tag{1}$$

In order to compute the Bayes probability, given any set of features as input, the gateway maintains a conditional probability table (CPT) as presented in Table 1.

Table 1. Example of a conditional probability table

	T = 1	T = 0
F_1	$p(FT = F_1 \vert T = 1)$	$p(FT = F_1 \vert T = 0)$
F_2	$p(FT = F_2 \vert T = 1)$	$p(FT = F_2 \vert T = 0)$
F_3	$p(FT = F_3 \vert T = 1)$	$p(FT = F_3 \vert T = 0)$

Each entry from Table 1 indicates the conditional probability of a mobile agent to deliver a sensing report which contains data with a given feature (e.g. temperature data), given a trustworthy transaction. According to Bayes formula, the entry from CPT can be computed following Formula 2:

$$p(FT=F_1|T=1)=\frac{p(FT=F_1,T=1)}{P(T=1)},\ \text{where} \tag{2}$$

$$p(FT=F_1,T=1)=\frac{\#\,of\ successful\ transactions\ with\ F_1}{\#\,of\ total\ transactions} \tag{3}$$

A transaction is classified as successful if its' degree of satisfaction passes a given threshold. This process is executed on the consumer application side by an evaluator agent that can contain a customized method of evaluation chosen by the user, thus it is considered out of the scope of this paper. For a static IoT device, a certain feature may be more important than others (e.g. receiving a high-quality temperature value may be more important than receiving an accurate air pressure value), thus the satisfaction degree formula allows assigning different weights to the evaluated features (as presented in Formula 4):

$$s=W_{F_1}\times S_{F_1}+W_{F_2}\times S_{F_2}+\ldots+W_{F_n}\times S_{F_n} \tag{4}$$

$$W_{F_1}+W_{F_2}+\ldots+W_{F_n}=1 \tag{5}$$

where W indicates the feature weight (importance) and S indicates a satisfaction value for a feature. If $S > S_t$ then the transaction is successful, otherwise it is unsuccessful.

Using Bayes theorem, the probability of a given mobile crowding IoT device to deliver a satisfying transaction that involves a feature set F is predicted.

$$p(T = 1|F) = \frac{\mathrm{p(F|T\ = 1) \times p(T\ = 1)}}{\mathrm{p(F)}} \tag{6}$$

When feature set F is expanded to features F_1, F_2, …, F_n, the Formula 6 becomes:

$$p(T = 1, F_1, F_2, \ldots, F_n) = p(T = 1) \times PROD\left(\frac{\mathrm{p\ (F}_i\mathrm{, T\ = 1)}}{\mathrm{p\ (T = 1)}}\right) \tag{7}$$

The naive Bayes algorithm implemented in the proposed framework provides a compact method of determining the reputation of data collected from crowdsensing devices, eliminating the risk of allowing nodes to inject malicious data into the IoT system.

3.3 Cloud Layer

In the proposed framework, the central position of the Cloud module empowers it to act as a management module and data relay for the entire IoT system. Considering the data relay role, the main task of the Cloud module is to relay sensing data requests coming from gateways. In order to do this, the request is first parsed and specific fields are extracted so that the request can be forwarded to a certain group of mobile crowdsensing IoT devices located in the proximity of the gateway that made the request. This is achieved by using the GPS location field found in the data sensing request. Furthermore, from this request the Cloud module will also filter the type of data the gateway requires, thus limiting the resources consumption from both implied parties (the crowdsensing IoT devices and the gateway).

Given that the mobile crowdsensing devices notify the Cloud module only when they connect to the network, it is difficult for it to have a real-time updated map of the entire network, but rather one that has the last status of each device. Therefore, several requests can be rejected, if the devices are not located in the targeted area, or discarded if the devices are not active anymore. In the first case, the crowdsensing devices send a message to notify the Cloud that their location has changed, while in the second case the Cloud module retries, for a customizable number of times, to send the request and, if no reply is received, it will mark the crowdsensing devices as inactive and remove them from further queries, until a reconnect message is received. Also, taking into consideration that these crowdsensing devices are mobile, some of them can move between areas of interest. In this case, the Cloud module will extend the area where the requests will be forwarded, so that any possible device that is currently active in the area of interest will be notified. Each communication link is secured using a symmetric key, randomly chosen by the Cloud module and specific for each crowdsensing device. For secretly sharing these symmetric keys with the corresponding crowdsensing devices, the Cloud module encrypts them with the public key of the crowdsensing devices.

Data gathered from the crowdsensing IoT devices groups, as producers, and used by the static IoT devices groups, as consumers, is trusted by the consumers in accordance with the reputation that the producers have. This level of reputation can fluctuate during the entire lifecycle of a producer and it can be used to detect malicious devices. Gateways can send reputation queries between them to see if a producer that crossed between areas covered by different gateways has been already assessed by the previous gateway and what is its level of reputation, or if it needs to be considered as a freshly registered producer and begin the reputation assessment process. Since gateways are manually registered by the administrator on the Cloud module, the setting of a trust relationship between different gateways is done automatically.

4 Implementation and Analysis

For the system implementation, we used Qemu for emulating the gateway and the static IoT devices, along with an Android application for the mobile crowdsensing. The static IoT devices logic was implemented as a Linux process that acts as an MQTT-SN client and communicates with the gateway for requesting data with the highest reputation. The aggregated data is published by the static IoT device to the gateway using MQTT-SN, the latter transporting the information to the consumer application through HTTPs (web service). For the mobile crowdsensing, we implemented a proof-of-concept Android application that communicates with the Cloud platform through Firebase messages (real-time push notifications). For the initial implementation we used only the smartphone onboard sensors along with software simulated sensors. We implemented a sensor abstraction layer to integrate the Android application with the simulated sensors, this abstraction layer allowing a rapid integration with a third-party wearable API.

For testing the naive Bayes reputation algorithm, we designed a custom Python simulator. The simulator allows declaring IoT nodes and associates different sensor types with the IoT node (e.g. an IoT node can deliver temperature and noise values). For each sensor type, a target value and a deviation interval were declared, this tuple being used to model the IoT node's behavior in a stochastic manner. For each sensor type we defined an evaluator model which gives a score (between 0 and 1) to each delivered data: if the data is accurate (close to the target value) the score is high. The evaluator model transmits the score to the naive Bayes engine that updates the reputation value on each simulation step. The goal of this experiment is to observe that an IoT node's reputation history is updated correctly by the naive Bayes engine based on the delivered data quality. In this experiment we used 3 sensors that deliver one or more data types. In the first test scenario, the sensors deliver temperature and humidity values: sensor 1 delivers the best values, followed by sensor 2 and sensor 3, as reflected in Fig. 3.

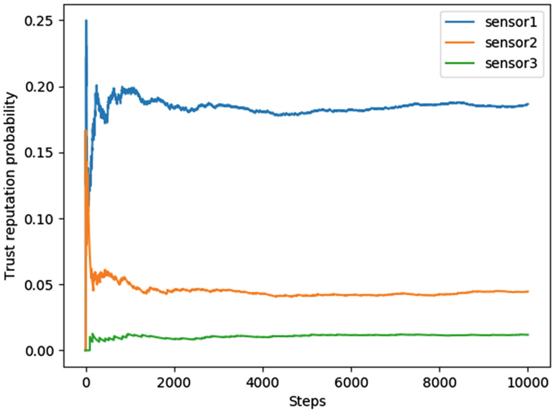

Fig. 3. First test scenario

In the second test case, the sensors deliver also temperature and humidity values: sensor 1 delivers the best temperature value and the second best humidity value, sensor 2 delivers the best humidity value and the second best temperature value, sensor 3 delivers the worst values. In this scenario, the humidity has a bigger weight (it is more important than the temperature value), as presented in Fig. 4.

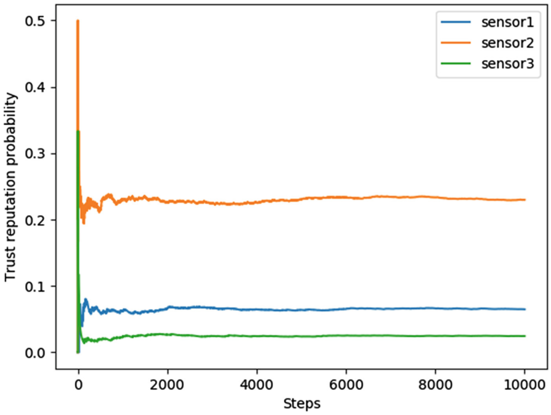

Fig. 4. Second test scenario

In the third test case, the sensors deliver temperature, humidity, and CO2 values: sensor 1 delivers the best values, followed by sensor 2 and sensor 3 for the first part of the simulation. For the second part of the simulation, sensor 3 delivers the best values, followed by sensor 2 and sensor 1, as presented in Fig. 5. This last test case simulates a data manipulation attack, where an IoT node achieves a high reputation score and then tries to manipulate the system by injecting false data.

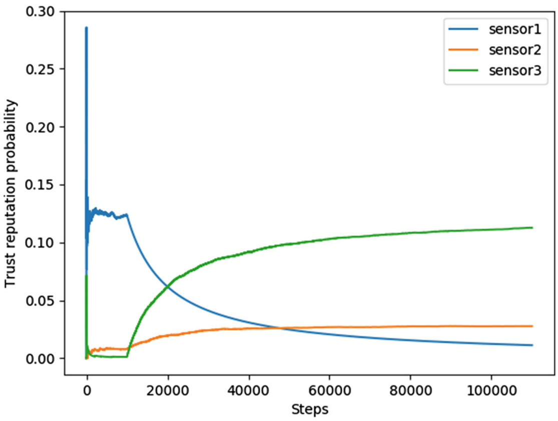

Fig. 5. Third test scenario

As the results for these three scenarios show, the reputation-based trust management system is able to adapt to changes and responds adequately to malicious intentions of pushing erroneous data into the IoT system. Also, as presented in the second scenario, if a weighted method of calculating reputation is chosen, the framework can cope with these changes and correctly assess the reputation of each node.

5 Conclusions

Mobile crowdsensing is trying to bring new data collection techniques into IoT by exploiting the sensing capabilities of users mobile devices to collect and share data. A major problem that arises in such applications is the impossibility of guaranteeing a suitable behavior for each mobile device. Hence the need for a security framework based on reputation, so that mobile device intervention with suspicious behavior can be minimized.

In this paper, we presented an approach to this problem by proposing a modular security framework able to compute the level of trust of a mobile device based on the feedback received from the consumer. A drawback of the model used in the decision-making process in the reputation system is that we use a threshold value that has to be set according to each type of application.

Regarding our future work, to prevent the aforementioned drawback, we plan to implement and test several reputation calculation algorithms in order to offer a trade-off between the algorithm accuracy and the required computing resources. By implementing a suite of algorithms either using Fuzzy logic, Case-Based Reasoning, or even naive Bayes, we can approach distinct IoT interaction models so that we can choose the right method of calculating reputation depending on the type of application. Another direction that we will focus on consists of improving the mechanism that ensures the anonymity of the crowdsensing devices while maintaining the system's responsiveness in the event of the occurrence of untrustworthy actions.

Acknowledgments. This work was supported by a grant of the Romanian Ministry of Research and Innovation, CCCDI – UEFISCDI, project number PN-III-P1-1.2-PCCDI-2017-0272/Avant-garde Technology Hub for Advanced Security (ATLAS), within PNCDI III.

References

1. Leonardi, C., Cappellotto, A., Caraviello, M., Lepri, B., Antonelli, F.: SecondNose: an air quality mobile crowdsensing system. In: Proceedings of the 8th Nordic Conference on Human-Computer Interaction, Helsinki, Finland, pp. 1051–1054 (2014)
2. Pan, B., Zheng, Y., Wilkie, D., Shahabi, C.: Crowd sensing of traffic anomalies based on human mobility and social media. In: Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Orlando, FL, USA, pp. 344–353 (2013)
3. Coric, V., Gruteser, M.: Crowdsensing maps of on-street parking spaces. In: Proceedings of the 9th IEEE International Conference on Distributed Computing in Sensor Systems, Cambridge, MA, USA, pp. 115–122 (2013)
4. Salpietro, R., Bedogni, L., Di Felice, M., Bononi, L.: Park Here! a smart parking system based on smartphones' embedded sensors and short range Communication Technologies. In: Proceedings of the 2015 IEEE 2nd World Forum on Internet of Things, Milan, Italy, pp. 18–23 (2015)
5. Ganti, R., Ye, F., Lei, H.: Mobile crowdsensing: current state and future challenges. IEEE Commun. Mag. **49**(11), 32–39 (2011)
6. Guo, B., Yu, Z., Zhang, D., Zhou, X.: From participatory sensing to mobile crowd sensing. In: Proceedings of the 12th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshop), Budapest, Hungary, pp. 593–598 (2014)
7. Giannetsos, T., Gisdakis, S., Papadimitratos, P.: Trustworthy people-centric sensing: privacy, security and user incentives road-map. In: Proceedings of the 13th Annual Mediterranean Workshop on Ad Hoc Networking, Piran, Slovenia, pp. 39–46 (2014)
8. Gunasekaran, S., Rathnamala, J.: Review on various architectural models in mobile crowdsensing (2015)
9. Montori, F., Bedogni, L., Di Chiappari, A., Bononi, L.: SenSquare: a mobile crowdsensing architecture for smart cities. In: 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT), Reston, VA, pp. 536–541 (2016)
10. Zheng, Y., Peng, Z., Athanasios, V.: A survey on trust management for Internet of Things. J. Netw. Comput. Appl. **42**, 120–134 (2014)
11. Ud Din, I., Guizani, M., Kim, B.-S., Hassan, S., Khan, K.: Trust management techniques for the Internet of Things: a survey. IEEE Access **7**, 29763–29787 (2018)
12. Chettri, R., Pradhan, S., Chettri, L.: Internet of Things: comparative study on classification algorithms (K-NN, naive Bayes and case based reasoning). Int. J. Comput. Appl. **130**, 7–9 (2015)
13. Chen, D., Chang, G., Sun, D., Li, J., Jia, J., Wang, X.: TRM-IoT: a trust management model based on fuzzy reputation for Internet of Things. Comput. Sci. Inf. Syst. **8**, 1207–1228 (2011)
14. Chen, I.R., Guo, J., Bao, F.: Trust management for SOA-based IoT and its application to service composition. IEEE Trans. Serv. Comput. **9**(3), 482–495 (2016)

15. Bao, F., Chen, I.: Trust management for the Internet of Things and its application to service composition. In: 13th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, San Francisco, CA, United States, pp. 1–6 (2012)
16. Nitti, M., Giran, R., Atzori, L., Iera, A., Morabito, G.: A subjective model for trustworthiness evaluation in the social Internet of Things. In: 2012 IEEE 23rd International Symposium on Personal Indoor and Mobile Radio Communication, Sydney, Australia, pp. 18–23 (2012)
17. Liu, T., Guan, Y., Yan, Y., Liu, L., Deng, Q.: A WSN-oriented key agreement protocol in Internet of Things. In: 3rd International Conference on Frontiers of Manufacturing Science and Measuring Technology, LiJiang, China, pp. 1792–1795 (2013)
18. Martinez-Julia, P., Skarmeta, A.F.: Beyond the separation of identifier and locator: building an identity-based overlay network architecture for the Future Internet. Comput. Netw. **57**(10), 2280–2300 (2013)