Crowdsourcing Under Attack: Detecting Malicious Behaviors in Waze

Luis Sanchez, Erika Rosas, Nicolas Hidalgo

Crowdsourcing under Attack: Detecting Malicious Behaviors in Waze

Luis Sanchez[1], Erika Rosas[2], and Nicolas Hidalgo[3]

[1] Departamento de Ingeniería Informática
Universidad de Santiago de Chile
luis.sanchezgu@usach.cl
[2] Departamento de Informática
Universidad Técnica Federico Santa María
erosas@inf.utfsm.cl
[3] Escuela de Informática y Telecomunicaciones
Universidad Diego Portales
nicolas.hidalgoc@mail.udp.cl

Abstract. Social networks that use geolocalization enable receiving data from users in order to provide information based on their collective experience. Specifically, this article is interested in the social network Waze, a real-time navigation application for drivers. This application uses methods for identifying users that are open and free, where people are able to hide their identity by using a pseudonym. In this context, malicious behaviors can emerge, endangering the quality of the reports on which the application is based. We propose a method to detect malicious behavior on Waze, which crawls information from the application, aggregates it and models the data relationships in graphs. Using this model the data is analyzed according to the size of the graph: for large interaction graphs, we use a Sybil detection technique, while for small graphs we propose the use of a threshold-based mechanism to detect targeted behaviors. The results show that it is complex to use the large-scale Sybil attack detection techniques due to parameter tuning. However, good success rates can be achieved to tag users as honest and malicious if there are a small number of interactions between these groups of users. On the other hand, for small graphs, a straightforward analysis can be performed, since the graphs are sparse and the users have a limited number of connections between them, making clear the presence of outliers.

Keywords: Online Social Networks; Malicious Behaviors; Sybil attack; Sybil detection

1 Introduction

In recent years, Online Social Networks (OSN) usage has increased exponentially and some of them have become part of most people's everyday life. Facebook[4],

[4] www.facebook.com

Instagram[5], Twitter[6] and Waze[7] are only a few examples of OSNs accounting for millions of users that interact daily creating and sharing information. In large OSNs, threats are just around the corner, not only menacing users' private data, but also the whole network goals. Identity theft, malwares, fake profiles (or Sybils) are common examples of threats present in this type of networks [8]. In this work, we are particularly interested in Sybil-based attacks under the well-known OSN for drivers, Waze.

Waze is a crowdsourcing application to assist drivers by providing online information on traffic and road conditions. It was created in 2008 and to-date, it has approximately 50 million users [10]. Waze creates an online report of traffic conditions for a given route based on the information collected from and reported by users: current speed, position, origin and destination, police controls, traffic jams, accidents, etc. One of Waze's main features is user engagement to contribute to the common good, i.e., Waze is not just crowdsourcing, but personal participation [10]. Waze's success is directly related to the good-will of users; therefore, malicious behaviors such as Sybil attacks can seriously compromise the application's precision and success.

In the last years, some of these attacks have been reported in [1, 13]. In the former, researchers were able to generate fake users and mobile devices using virtual machines in Android and they created fake traffic jams by setting low speed configurations to their fake devices. In the latter, people coordinated to emulate a traffic jam in a residential area, in order to reduce car passing in their neighborhood. Attacks like these can seriously compromise the behavior of the application and detecting this can be challenging in the presence of millions of users dynamically interacting online with the presence of anonymous users in the system. Sybil detection in similar environments such as Twitter have been studied; however, in the context of Waze, the application of state-of-the-art Sybil detection mechanisms requires modelling the malicious behaviors in terms of the data collected from the network.

In this work, we propose three models that attempt to characterize three different behaviors on which we focus our study: (1) Collusion for traffic jams: people collude in order to simulate a traffic jam by not moving [1]; (2) Driver speed attack: a coordinated group of Sybils simulate slow driving so that Waze declares a false traffic jam [13]; and (3) False event attacks: a coordinated group of Sybils vote for a false event, that obscures honest users.

The main contributions of this work are the models associated to the three malicious Sybil behaviors previously exposed. Our models were tested with real Waze traces and our results that show that malicious behaviors can be detected using a state-of-the-art Sybil attack detection mechanism and a threshold-based mechanism. In our experiments, we have exploited SybilDefender [14] and a threshold-based mechanism to detect abnormal behaviors. The former is applied over large interaction graphs and the latter, over small ones.

[5] www.instagram.com
[6] www.twitter.com
[7] www.waze.com

The remainder of this work is organized as follows: Section 2 briefly introduces the identity problems and their relationship with Sybil attacks. Also, we present some literature works that attempt to tackle this problem. Section 3 presents the main contribution of this work, the graphs that model the malicious behaviors we attempt to detect. In Section 4, we evaluate the proposed malicious behavior models using SybilDefender as the mechanism to detect Sybils and a threshold-based mechanism for those small interactions graphs. Finally, our conclusions and future work is stated in Section 5.

2 Background and Related Work

2.1 Identity Attacks in Social Networks

An identity in a social network is the set of characteristics of a particular person or group (entity) that distinguishes it from others in the network. Different to real-life identities, such as identification cards or a passport that are shown by a person and that can be confirmed comparing a picture and the biometric indexes, in the online world it is more difficult to establish the link between a physical entity and the online identity that represents it.

This problem has been widely discussed because it is easy to change one's identity in several OSNs, whereas in real life, this is a complex process. Friedman and Resnick have called this type of identity *cheap pseudonyms* [9], and allows a person interacting anonymously to constantly change identifiers or to maintaining a persistent identity.

In this context, one unique entity can build a set of pseudonyms in the system, which makes it appear as different entities. What we call a Sybil attack occurs when one physical entity creates and uses a set of identities in the system in order to perform malicious behaviors [6]. The malicious behaviors may vary according to the online environment and it may range from exploiting more resources than allowed to performing active attacks that hamper the veracity of the information exchanged in the network.

The problem of cheap pseudonyms is that they reduce the number of accountable actions in the system, and in the case of Waze, one user that has multiple identities in the system or multiple users may collude to spread false information for other drivers. This network shows relevant information to the users based on their localization and false information may show false traffic jams, which may produce longer routes for other drivers.

This kind of problem has been studied by [17] in the context of the social network Dianping. They have found that some user accounts make positive comments about places that are very far from each other in time intervals that are impossible to achieve. A few users control these accounts that give good rating to some places and bad rating to the competence, in exchange for money.

In the case of Waze, Sinai et al. [13] coordinated a Sybil attack by creating multiple identities using multiple virtual machines in Android that ran the Waze application. They simulated slow driving in all the identities on a specific street

so the system detected a false traffic jam. They have proved that it is possible to control routes, which may produce important problems for other drivers. A collusion attack has also been documented by Carney [1] in LA, USA. In this case, the neighbors colluded and activated Waze outside their houses in order to simulate a false traffic jam, in order to force Waze to not recommend that neighborhood to drivers.

2.2 Managing Sybil Attacks and Collusions

In the context of large-scale systems, we can find two types of approaches to prevent the Sybil attack and collusion: detection and tolerance. The *detection* of malicious behaviors is focused on detecting identities that are acting with malice in the system, and evict them from the system. However, in presence of cheap pseudonyms, there is no problem for them to obtain a fresh new identity. For this reason, in large-scale networks, a more common approach is to *tolerate* malicious behaviors, for example, by avoiding using information generated by suspicious identities.

Community detection techniques have been used to detect suspicious entities in the presence of the Sybil attacks, assuming that the number of interactions of these types of identities with real users is limited, that there is at least one honest known user in the network, and the honest region is densely connected. Several Sybil detection algorithms have been proposed [4, 14, 16] that classify identities as Sybils or normal.

In this work, we used SybilDefender [14], but any other Sybil detection system can be used in its place. SybilDefender proposes four algorithms; the first one obtains statistics from the neighborhood of an honest node identifying J judges from its vicinity and performing R random walks of length L. The second one identifies a suspicious node as Sybil or non-Sybil using the results of the first algorithm. This is performed through R random walks of length L from the suspicious node, and comparing the values of recurrent nodes in the results of algorithm 1. The third and fourth algorithms enable detecting a Sybil region around a node classified as Sybil. A detailed description of the algorithms is presented in [14].

Recently, Sybil attacks have been studied in the context of Vehicular Ad-hoc Networks (VANETs) [7, 11, 12]. In [7], the authors built an event-based reputation system that feeds a trust management system that restricts the dissemination of false messages. In [11, 12], the authors use driving patterns of vehicles and detect Sybils using classifiers, such as minimum distance classifier and support vector machines. In location-based social networks, [15] states that it is not normal the appearance of continuous gatherings, and use the detection of these events to identify Sybils. However, we argue that traffic jams may produce continuous gathering in urban zones. Other graph-based solution has been proposed for mobile online social networks in [3], where the authors use a connection analysis to differentiate honest versus fake nodes. Finally, in the context of mobile crowdsourcing, recent work proposes a passive and active checking scheme that

verify traffic volume, signal strength and network topology [2], differentiating nodes using an adaptive threshold.

3 Modeling Sybil behaviors

The goal of the model is to identify Sybil attacks in Waze. The key contribution is the way of modeling the data of Waze in order to detect Sybils or collusion. Figure 1 shows the proposed pipeline that is detailed in the following subsections. In general, data is captured from the LiveMap of Waze, reordered, and aggregated in order prevent redundant information. Then, we generate graphs that model the interactions between the users according to the malicious behavior we target. Then, an analysis of the graphs is performed using state-of-the-art Sybil detection mechanisms.

3.1 Crawling and indexing

We crawled data from Waze using the endpoint that feeds the LiveMap[8] of Waze. This is public data that is delivered in JSON format of the current state of the requested area, defined by coordinates. The data captured was requested every 1 minute, obtaining a snapshot of the map at that time. We categorize data in three types:

- **Alerts**: Alerts are events explicitly reported by Waze users, such as vehicle accidents, police locations, traffic jam reports, etc. These alerts are characterized by a point on a map and the number of votes received by other users that corroborate the information.
- **Jams**: Traffic jams are events that Waze reports using the location data of users. They are created when large traffic is detected on a street and are represented by geographic coordinates that build a line, and other data such as the severity of the traffic jam and current user speed, among others.
- **Users**: The data of users on the map shows their current location and is represented by an identifier, geographic coordinates, and current speed, among others.

We have indexed the data by user identifier and event identifier. The information about traffic jams is not used in this study, since it does not contain user information, which is the main focus of this study.

3.2 Data Aggregation

We mainly produce two structures that facilitate graph modeling:

[8] `http://www.waze.com/livemap`

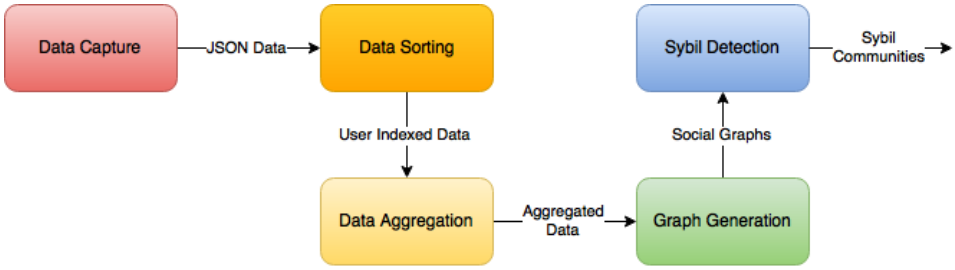

Fig. 1. Overview Steps of the Solution

- **Event fusion**: Generally, users create several Waze alerts that correspond to one real traffic event. In order to build the relationship between users that interact with a real event, we fuse the votes that are close in time and space. We have set the time and space parameters for 30 minutes and 200 meters respectively.
- **Routes generation**: In order to relate users to a route trip they have followed, we have generated routes from the geographic coordinates of the users. The routes are built from the user geographic points that are relatively close in time. Since the data is captured every 1 minute, it is not easy to generate a route from two geographically different points. We have used the process called map matching first, in order to deal with vagueness of the GPS location, so that the points are inside a street. Then, we have used Google Roads API, to correct the coordinates and infer the route that the user took.

3.3 Graph modelling

As we attempt to exploit state-of-the-art Sybil detection mechanisms, we must model the data associated to the different malicious behaviors that we attempt to detect. In this work, we focus on three problems, some of them already reported in literature. The malicious behaviors we target are:

- **Collusion for traffic jam:** A group of users is detained on a street so that Waze declares a false traffic jam. This behavior is described in [1].
- **Driving speed attack:** A coordinated group of Sybils that simulate slow driving so that Waze declares a false traffic jam. This behavior is described in [13].
- **False event attacks**: A coordinated group of Sybils that vote for a false event that obscure the honest users.

The generated interaction graphs are non-directed defined as $G = (V, A)$, where V is the set of vertexes that represent users, and A indicates the set of edges that represent interactions between two users. The weight of the edge is related with the amount of interactions the users had in time.

If the modeled graphs are fast mixing and the number of connections from the malicious to the honest area is limited, then, we can apply a Sybil Detection algorithm. With these properties, we guarantee that random walks can iterate all over the honest area, and it is difficult to walk outside this area.

Traffic jam graph This graph is built with the users that do not move at the same time and within a close distance. Then, an interaction between two users a, b is defined as the number of times they were standing still at the same time in any of their trips v. Equation (1) shows how the weight of an edge between two users is computed: the sum of the number of times they may be colluded. ST is the set of trips where the users had the same origin and destination, meaning that they did not move from the beginning of the trip.

Equation (2) shows when a possible collusion is detected: when in a time window t the users were within a distance of less than d meters. We consider that the function $distance(v_1, v_2)$ gives the distance in meters from the position of the trip v_1 to the position of the trip v_2 and $time(v)$ gives the time when the travel v happened.

$$w_{g1}(a,b) = \sum^{v_{ua},v_{ub} \in ST} col(v_{ua}, v_{ub}) \tag{1}$$

$$col(v_1, v_2) = \begin{cases} 1, & \text{if } distance(v_1, v_2) < d \text{ and} \\ & |time(v_1) - time(v_2)| < t \\ 0, & \text{in any other case} \end{cases} \tag{2}$$

If the users collude to stand still in the streets in order to produce a traffic jam, then they will appear more connected in the graph, and we would like to detect this malicious area.

Driving speed graph The goal of this graph is to detect inconsistencies in the user behaviors that share part of their travel routes. We compare speed and temporality in order to check if their characteristics validate each other. The weight of an edge in this graph is high when two users share routes close in time and their driving speeds were similar. A lower weight of an edge indicates that their driving speeds were very different. Equation (3) shows the weight of the edge between users a and b: In the sum of all the times there is a similarity found between their trips T, normalized by the minimum number of trips of the users. We call v_{ua} a trip of the user a and v_{ub} a trip of the user b.

Equation (4) shows what we consider a similarity in this case. We take into account three properties of the trips, the time, the routes and the speed. Route similarity $s_{routes}(v_1, v_2)$ computes the number of segments shared in the trip v_1 and v_2, divided by the number of segments of the longest trip between v_1 and v_2. This route similarity is modified by factor α and β so to favor a homogeneous distribution, incrementing the similarity to 1 in cases there is a high similarity in

speed and time, and generating a medium effect when there is a regular similarity in speed and time.

The temporal similarity $s_{time}(v_1, v_2)$ is equal to zero if v_1 and v_2 were separated in time more than a value min and equal to 100% if v_1 and v_2 occurred at the same time. Temporal distances in between are proportionally computed. The speed similarity $s_{speed}(v_1, v_2)$ indicates whether the segments that share v_1 and v_2 have a similar speed. This is computed with the average speed of these segments.

$$w_{g2}(a,b) = \frac{\sum^{v_{ua}, v_{ub} \in T} sim(v_{ua}, v_{ub})}{min(\cup_{i=a,b} \text{number of user trips } i)} \tag{3}$$

$$sim(v_1, v_2) = \begin{cases} s_{routes}(v_1, v_2) \times \alpha, & \text{if } s_{speed}(v_1, v_2) > max \\ & \text{and } s_{time}(v_1, v_2) > max \\ s_{routes}(v_1, v_2) \times \beta, & \text{if } s_{speed}(v_1, v_2) > min \\ & \text{and } s_{time}(v_1, v_2) > min \\ s_{routes}(v_1, v_2), & \text{in any other case} \end{cases} \tag{4}$$

$$\alpha = \frac{1}{max(\cup_{i=0}^{n} w_{g2})} \tag{5}$$

$$\beta = 1 + (\frac{1-\alpha}{2}) \tag{6}$$

In extreme cases of malicious behaviors, the users that share temporality and space will be strongly connected. Honest users would have low weighted edges.

False event graph The main goal of this graph is to identify user groups that vote on the same events. In this case, the weight of each edge $w_{g3}(a, b)$ that links the users a and b is the sum of the events that the users a and b have voted for the same event.

Malicious users that also vote for events that honest users have voted on may create a graph that is not fast mixing and hinder Sybil detection. However, they require the effort of creating strong links with other users. Figure 2 shows a random subgraph built with the experimental data. Each vertex represents a user and the links represent the interaction the users had in time. The weight of these edges is determined by the number of votes of the users on the same events.

3.4 Malicious Behavior Detection

The malicious behavior was detected by analyzing the graphs and applying a threshold based approach or a Sybil detection algorithm, according to the characteristics of the graph. We will tag a user as malicious when:

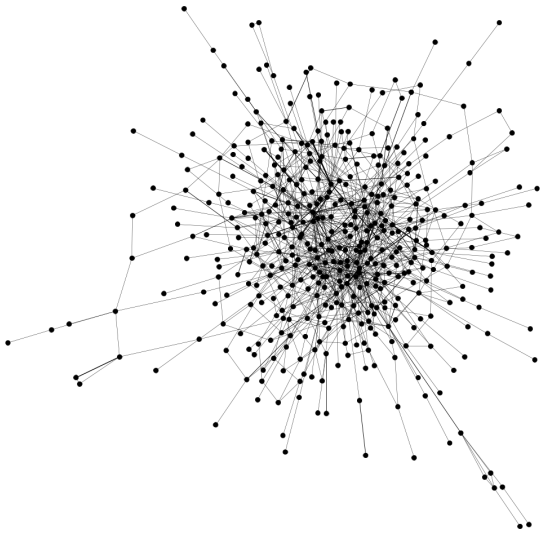

Fig. 2. Events graph

- In the first graph, the more connected regions will be the malicious areas. Honest users that do not collude with others should present a small number of connections.
- In the second graph, strong relations are given when users share their behavior, which can be honest or malicious. We have to start the process with a previously known honest user to identify the regions.
- The third graph is similar to the second; we assume that users do not vote in group for an event, so the number of connections between them are going to be smaller compared to malicious Sybil users.

4 Experimentation

The data was crawled in a timespan of 6 months, from July 2015 until January 2016, with some data missing in October 2015 due to a server failure. We used the LiveMap API of Waze that was consulted every 1 minute. The area consulted was around the city Santiago, Chile with the coordinates -33.2 North, -33.8 South, -70.87 East, -70.5 West.

In total, we considered $1,667,400$ events, that where generated by $223,031$ users. We captured $4,547,887$ users with locations inside the coordinates, which is a large number, considering the size of the city of Santiago; however, Waze uses new identifiers for anonymous users, which explains the number of users.

In the data aggregation step, 30% of the events were fused. Figure 3 shows the distribution of the events per hour of the day. We are able to observe the

peak time of the day in the figure, which is normal for a city like Santiago, at times when people go to work and when they return home.

In the data aggregation step, we have also obtained user trips with the trace we built from their locations that are close in time (maximum 7 minutes of difference between each subsequent pair). We consider trips that have at least 5 consecutive location points. These values were experimentally chosen, since most of the trips have an average of 1 minute between consecutive points. The result is $192,248$ trips of $184,992$ users (mostly anonymous users). Figure 4 and 5 show the distribution of the duration and the distance traveled in each trip. Most trips are short in time and distance.

4.1 False Event Graph

The obtained graph has $223,030$ vertexes and $1,452,261$ edges, with an average degree of 13. This is a non-connected graph, and has $59,815$ components. In order to perform the experiments we selected the largest component of the graph that has $160,894$ vertexes and $1,499,717$ edges. The remaining graphs have a size that is negligible for this experiment.

In order to obtain a fast mixing graph, we removed vertexes with small degrees. Figure 6 shows how the mixing time changes with the minimum degree of the graph. This means that we can apply a Sybil detection mechanism when the graph is dense, with a minimum degree of 64. The resulting graph has $9,743$ vertexes and $720,152$ edges with an average degree of 73.

We connected a Sybil area to this graph in order to observe how the algorithm behaves. The Sybil area was created using the Erdõs-Rényi model to create random and sparse graphs. This model is often invoked to capture the structure of social networks [5] and is defined as a set of N nodes connected by n edges chosen random from the $(N(N-1))/2$ posible edges.The Sybil area had 523 users and $1,080$ edges, with a mixing time of 41. The number of vertexes corresponds to 0.03% of the number of honest nodes in the event graph. To connect both regions we created random attack edges between each region.

SybilDefender works mainly with two algorithms: the first identifies J judges from an honest known node. Then, from each judge, R random walks are computed of length L, counting how many nodes appear more than T times in the random walks. Then, the average and the standard deviation are computed for each element of the list L. In our case, we set the number of judges to $T = 50$, the number of random walks to $R = 100$, $L = 100, 200 \ldots 1000$ and $T = 5$, considering the size of the experiments presented in [14].

The second algorithm identifies if a node is Sybil or not, using the results of the first algorithm. This is done computing R random walks of length L from the suspicious node, counting how many nodes exceed T repetitions, which is called m. Then, a comparison is done to determine if the node is Sybil or not: $media - m > deviation \times \alpha$.

SybilDefender, with 20 attack edges joining the Sybil and honest areas, found a 99.1% of successfully detected Sybils and 100% successfully detected honest users using SybilDefender algorithms. However, if the number of attack edges

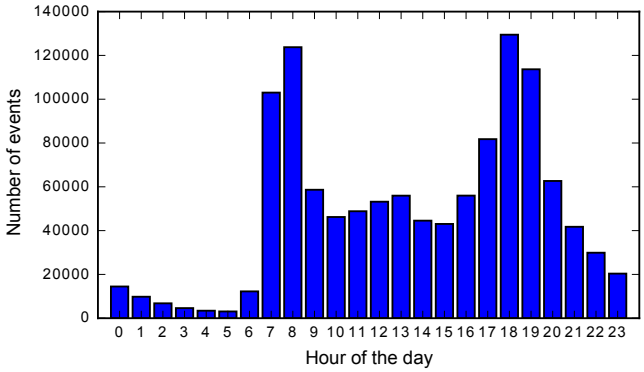

Fig. 3. Events per hour of the day

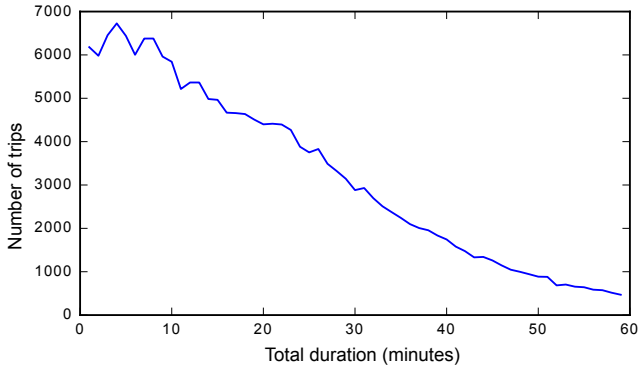

Fig. 4. Distribution of total trip time

grows to 40, then there is a 19% of successfully detected Sybil users and 100% of successfully detected honest users.

We modified the parameters of the algorithms, in order to observe the influence on the results. When using a $T = 7$ and changing the value α we obtained the results of Figure 7. The best results are obtained when using $\alpha = 55$, when we obtained a 76% of Sybils detected and 96% of honest users.

Furthermore, if we modify the length of the random walks, we obtain the results of Figure 8. In this case, the algorithm always detects all the honest users and the best case for Sybils occurs at 350 maximum length where it finds 83% of Sybils.

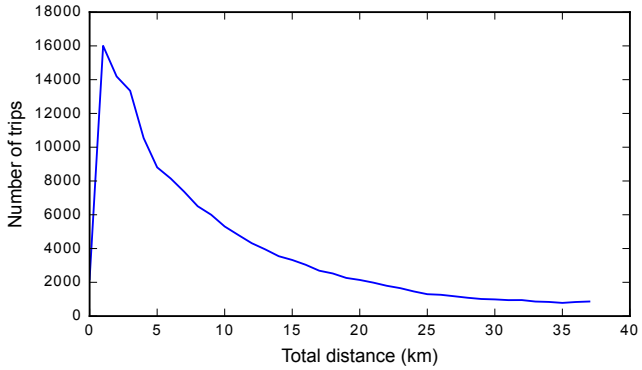

Fig. 5. Distribution of total distance of trips

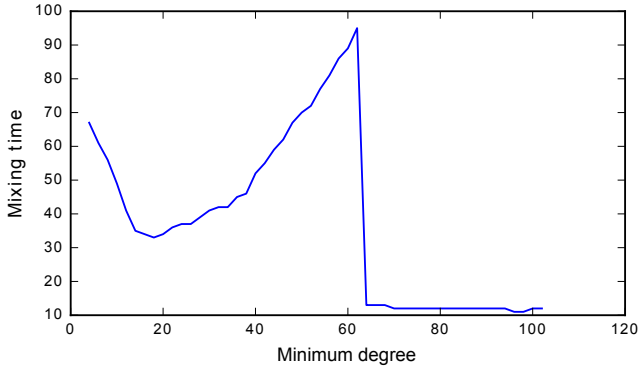

Fig. 6. Graph mixing time when vertex are removed

4.2 Driving Speed Graph

This graph was built with parameters min and max set on 30 and 70, respectively. The resulting graph has 149, 492 vertex and 7, 048 edges. This is a non-connected graph with 142, 582 components. Unlike the previous graph, this one has 138, 810 components with one vertex, 2, 479 with two vertexes and 1, 293 with three or more vertex, with the largest component with 24 vertexes. The size of the graph is too small to apply Sybil defender.

Figure 9 shows the value of the edges in the largest component. Analyzing the edges values, we located an edge with a similarity value of 44,51. This is because they had a very different speed in the shared segments of the trip.

We have drawn all the travels involved in the generation of the graph, and obtained Figure 10. It is clear that the graph show honest users that have similarities in the highways of the city.

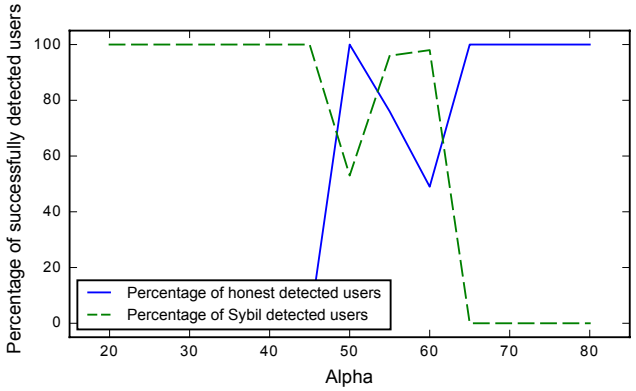

Fig. 7. Sybil defender results modifying α values

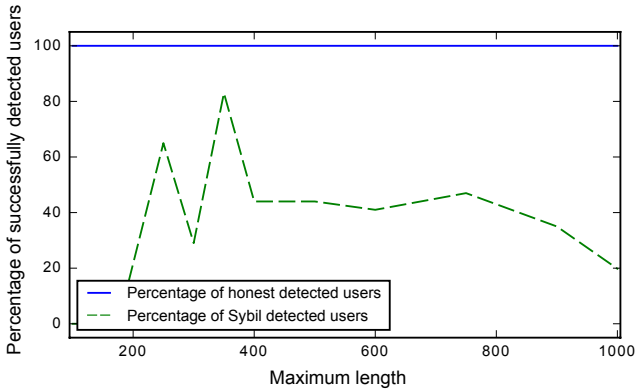

Fig. 8. Sybil defender results modifying maximum length

4.3 Traffic Jam Graph

In this case, the parameter d was set to 1km and t equals to 30 minutes. With this parameters we obtained a graph for this case has $38,455$ vertex and $8,592$ edges, with an average degree of 0.44 (maximum degree 14). This is a non-connected disperse graph that has $31,967$ connected components. The largest part has 711 vertex and $1,448$ edges, which is small for applying Sybil defender. Figure 11 shows the locations of the users of the graph, mostly located in the northeastern area of Santiago.

We analyzed the distribution of the edges weight and found that from 711 users, there are 643 that have only one stand still trip, 58 have only two stand still trips, 9 have between 3 and 8 trips and only one user have 27 stand still trips. Thus, we conclude that colluded users may be detected by their degree in the graph, which in our case can be set around 10 to consider a user as suspicious.

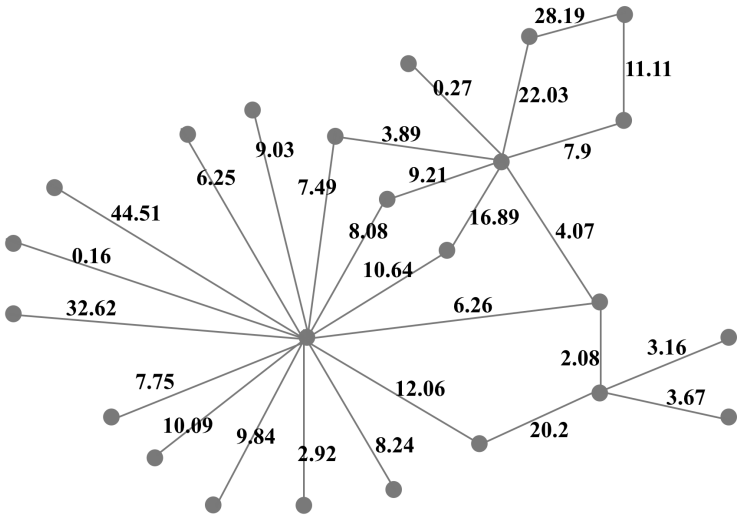

Fig. 9. Largest component of driver speed graphs

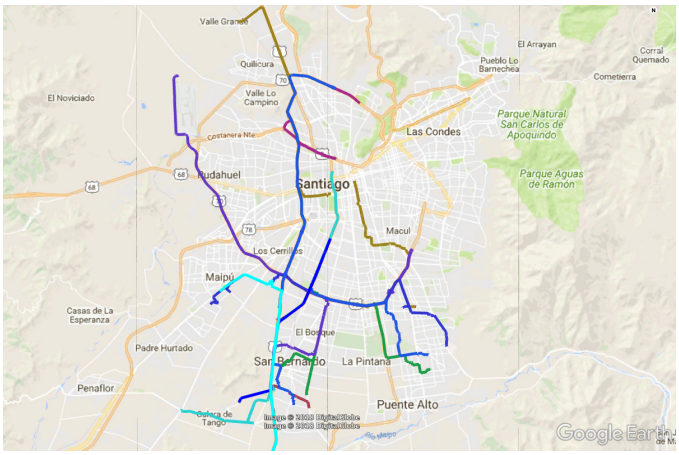

Fig. 10. All the routes of the largest component of the driver speed graph

5 Conclusion

In this work, we have proposed a model to process Waze traces and detect Sybil behaviors. The model consists of five steps, standing out the modeling of targeted behaviors on an interaction graph. We provide three of these models that characterize the three different behaviors which we focus on in our study: (1) Collusion for traffic jam, where people collude in standing still to simulate a traffic jam and divert traffic; (2) Driver speed attack, where a coordinated group of Sybils simulate slow driving to trick the application into assuming a traffic

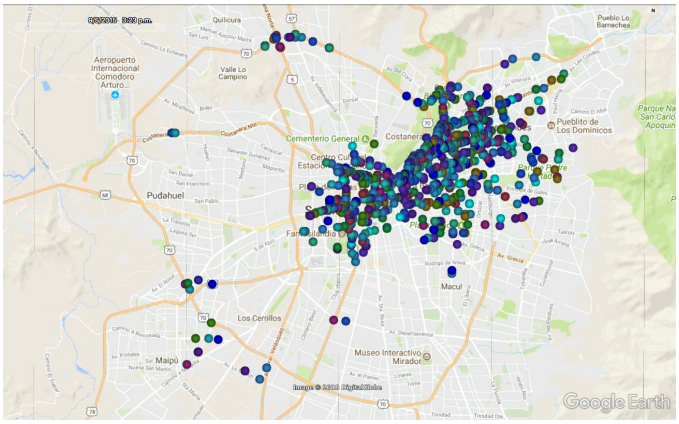

Fig. 11. Location of users of traffic jam graph

jam; and (3) False event attacks, where a coordinated group of Sybils vote for a false event that obscures the honest users.

The general model was tested with real Waze traces and our results that show that malicious behaviors can be detected using a state-of-the-art Sybil attack detection mechanism and a threshold-based mechanism. In our experiment, we have exploited SybilDefender [14] and a threshold-based mechanism to detect abnormal behaviors. The former is applied on large interaction graphs and the latter, over small ones where the application of a large-scale detection mechanism is not necessary.

Our results show that it is complex to use the large-scale Sybil attack detection techniques due to the parameter tuning. However, good success rates can be achieved to tag users as honest and malicious, if the number of interactions between those groups of users is small. On the other hand, for small graphs, a straightforward analysis can be performed since the graphs are sparse and the users have a small number of connections between each other, making clear the presence of unusual behaviors.

Acknowledgment

This work is partially supported by the University of Santiago research project PMI-USA 1204 and FONDEF Idea ID15I10560, CONICYT, Chile.

References

1. Carney, M.: Angry LA residents are trying to sabotage Waze data to stop side-street congestion (2014), `https://pando.com/2014/11/17/angry-la-residents-try-to-sabotage-waze-data-to-stop-side\-street-congestion/`

2. Chang, S.H., Chen, Z.R.: Protecting mobile crowd sensing against sybil attacks using cloud based trust management system. Mobile Information Systems 2016, 10 (2016)
3. Chinchore, A., Jiang, F., Xu, G.: Intelligent sybil attack detection on abnormal connectivity behavior in mobile social networks. In: Uden, L., Heričko, M., Ting, I.H. (eds.) Knowledge Management in Organizations. pp. 602–617. Springer International Publishing, Cham (2015)
4. Danezis, G., Mittal, P.: SybilInfer: Detecting Sybil Nodes using Social Networks. Tech. Rep. MSR-TR-2009-6, Microsoft (Jan 2009)
5. Dobrescu, R., Ionescu, F.: Large Scale Networks: Modeling and Simulation. CRC Press; 1 edition (2016)
6. Douceur, J.R.: The sybil attack. In: Revised Papers from the First International Workshop on Peer-to-Peer Systems. pp. 251–260. IPTPS '01, Springer-Verlag, London, UK, UK (2002)
7. Feng, X., Li, C.y., Chen, D.x., Tang, J.: A method for defensing against multi-source sybil attacks in vanet. Peer-to-Peer Networking and Applications 10(2), 305–314 (Mar 2017)
8. Fire, M., Goldschmidt, R., Elovici, Y.: Online social networks: Threats and solutions. IEEE Communications Surveys Tutorials 16(4), 2019–2036 (Fourthquarter 2014)
9. Friedman, E.J., Resnick, P.: The Social Cost of Cheap Pseudonyms. Journal of Economics & Management Strategy 10(2), 173–199 (2001)
10. Goel, V.: Maps that live and breathe with data (2013), http://www.nytimes.com/2013/06/11/technology/mobile-companies-crave-maps-that-live-and-breathe.html
11. Gu, P., Khatoun, R., Begriche, Y., Serhrouchni, A.: Vehicle driving pattern based sybil attack detection. In: 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS). pp. 1282–1288 (Dec 2016)
12. Gu, P., Khatoun, R., Begriche, Y., Serhrouchni, A.: Support vector machine (svm) based sybil attack detection in vehicular networks. In: 2017 IEEE Wireless Communications and Networking Conference (WCNC). pp. 1–6 (March 2017)
13. Sinai, M.B., Partush, N., Yadid, S., Yahav, E.: Exploiting Social Navigation. arXiv:1410.0151 [cs] (Oct 2014)
14. Wei, W., Xu, F., Tan, C., Li, Q.: SybilDefender: Defend against sybil attacks in large social networks. In: 2012 Proceedings IEEE INFOCOM. pp. 1951–1959 (Mar 2012)
15. Xu, Z., Chen, B., Meng, X., Liu, L.: Towards efficient detection of sybil attacks in location-based social networks. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI). pp. 1–7 (Nov 2017)
16. Yu, H., Kaminsky, M., Gibbons, P.B., Flaxman, A.: Sybilguard: Defending against sybil attacks via social networks. In: In ACM SIGCOMM '06. pp. 267–278. ACM Press (2006)
17. Zhang, X., Zheng, H., Li, X., Du, S., Zhu, H.: You are where you have been: Sybil detection via geo-location analysis in OSNs. In: 2014 IEEE Global Communications Conference (GLOBECOM). pp. 698–703 (Dec 2014)