Twitter Analysis for Real-Time Malware Discovery.

Article

Accepted version

F. Concone, A. De Paola, G. Lo Re, M. Morana

Proceedings of AEIT in Cyber Security:
International Annual Conference of AEIT, Cagliari, Italy, 2017.

Twitter Analysis for Real-Time Malware Discovery

Federico Concone, Alessandra De Paola, Giuseppe Lo Re, and Marco Morana*
DIID, Università degli Studi di Palermo, Italy
{federico.concone, alessandra.depaola, giuseppe.lore, marco.morana}@unipa.it
*corresponding author

***Abstract*—In recent years, the increasing number of cyber-attacks has gained the development of innovative tools to quickly detect new threats. A recent approach to this problem is to analyze the content of Social Networks to discover the rising of new malicious software. Twitter is a popular social network which allows millions of users to share their opinions on what happens all over the world. The subscribers can insert messages, called *tweet*, that are usually related to international news. In this work, we present a system for real-time malware alerting using a set of tweets captured through the Twitter API's, and analyzed by means of a Bayes Naïve classifier. Then, groups of tweets discussing the same topic, e.g, a new malware infection, are summarized in order to produce an alert. Tests have been performed to evaluate the performance of the system and results show the effectiveness of our implementation.**

***Keywords–Social Sensing*; *Twitter Analysis*; *Malware alerts*;**

I. Introduction

In recent years, we assisted at an increase in computer threats from the network, including the spread of malware. The main cause of this phenomenon is closely related to the technological development of communication systems. As a consequence, it is more and more necessary to introduce new techniques to prevent, detect and limit the cyber-attacks. In addition to traditional malware detection systems, new tools to boost and speed up the whole process have been proposed [1].

An effective way to create reliable solutions is to let new systems exploit the collaboration among multiple individuals, for example by analyzing data shared through the online social networks (OSNs). Monitoring and analyzing user reviews and opinions allow detecting trends, likes, and positive or negative factors related to each topic [2]. Sharing information between users makes social networks a powerful tool not only for updating other users in real-time about everyday life, but also for influencing people behaviors.

In the case of Twitter, the service offered to subscribers consists in sharing messages, called tweets, containing a maximum of 140 characters. Every day, 55 million messages are sent on average, with peaks touching 3,000 tweets per second in exceptional cases, usually related to new international events. Twitter is used both by corporations and people for several purposes, such as sharing news, opinions, rumors, publishing real-time information or announcements of new products, services, events and so on. Compare to other OSNs, Twitter has proven to be a very useful environment due to its popularity, that is users' tweets significantly help people to be promptly informed about what is happening in the world.

The most common limitation of systems addressing the issue of content-based social network analysis is the difficulty in performing real-time detection [3]. A quick analysis requires to download a set of tweets and analyze them within a certain temporal window. Most of the solutions are based on fixed-size windows, thus they are frequently not able to process and then report an event in a timely manner.

This paper presents a novel system aimed at creating automatic alerts about new malware spreads in real time. To this end, the proposed solution is based on variable-length windows, whose size is modeled through a sigmoid function. This allows to detect an event anytime the number of tweets exceeds the sigmoid value at time t.

The remainder of the paper is organized as follows. Section II briefly introduces some related works focusing on social network analysis for topic detection and malware discovering, while Section III illustrates the proposed architecture for real-time malware alerting. Section IV will present the experimental results obtained by the system we propose here. Conclusions follow in Section V.

II. Related Work

The term "Malware" refers to different types of threats, including viruses, worms, backdoors, zombies, trojan horses, that can spread out in different ways, e.g., e-mail, websites, portable storage drives and so on [4]. Malware propagation in social networks is still an open research field.

In order to face such threat, several works focus on social network analysis and differ according to the approach they adopt. Most of these works analyze social structures and the activities performed by users. For example, in [5] is described a system that analyzes the social graph formed by users and the user's activities. Another work is presented in [6], where the authors propose a worm detection system, which considers both the propagation of worms and the topological properties of social networks. Unfortunately, such systems have several limitations because they are based on some underlying assumptions. An example is presented in [7], where the authors assume that a malware tries to infect all the users that are online in a certain time. However, as discussed in the same work, this propagation model is not general and so the applicability of the system is quite limited.

Among existing online social networks, Twitter is one of most relevant to malware detection purpose, because it exhibits several advantages compared to other social media. The most important feature is that all users, even non-registered, can view tweets shared by other users without any link between the two entities. For this reason, more and more researches focused on the analysis of the huge amount of open information coming from Twitter so as to find events and topics of interest [8]. In [9] a real-time system, based on wavelet analysis of hashtags occurrences and topic summarization is presented. Another framework is described in [2], [9], where Twitter stream is processed to perform real-time event detection focusing on the user's perspective. A notable feature of such system is the adoption of dynamic length windows that, basing on a sigmoid function, permit a faster recognition of events than other systems based on static length windows.

Some other works focused on Twitter analysis in order to find malware, malicious accounts [10], mass emergency events [11]. An example is *WarningBird* [12], a robust system that uses Twitter stream in order to detect suspicious URLs. In particular, such framework assumes that attackers reuse the same URLs to deceive the victim and, so, it analyzes the correlations of URL redirect chains and determines their suspiciousness. The authors of [3] present an automatic system that analyzes some Twitter posts in real-time and creates alerts if the number of posts, containing a particular set of keywords, exceeds a threshold value. Unfortunately, this system presents some issues. Two of the most important limitations are the inability to distinguish two or more malware spreads occurring at the same time interval, and the static and predetermined length of time windows. Moreover, the whole alerting process depends on maximum and minimum thresholds to exceed in order to detect an event, which is unpractical when dealing with events which involve a variable volume of tweets.

III. Architecture

The architecture we propose here is able to analyze Twitter posting, identify the spread of a relevant event, and report the event itself. Specifically, the proposed system uses information contained in tweets that are published when a virus or malware is detected by users, or when a security report is release by certified authorities. The architecture can be summarized by the block diagram of Fig. 1. The system extracts users post from Twitter streaming through the Twitter APIs. All tweets, containing a set of selected keywords, i.e., words that are frequently found in messages related to computer attacks, are processed in real-time to detect the spread of new malware.

Since a tweet containing a keyword is not always closely related to the spread of new malware on the network, e.g., some tweets may refer to anti-virus advertisements, each tweet is pre-processed as follows. All "not important" words are deleted from the tweet collection to support the classification process, then tweets are filtered by means of a Bayes Naïve classifier, which was previously trained and whose purpose is to select messages that represent news about new cyber-attacks and malware.

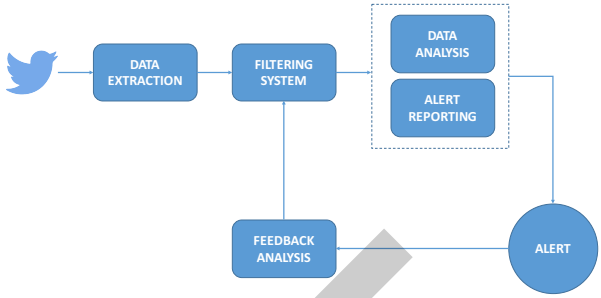

Fig. 1. System architecture.

The following of this section describes the different components of the alerting architecture.

A. Real-time data extraction

Streaming APIs, provided by Twitter, are used to collect in a continuous way all tweets containing a set of keywords referring to viruses or malware [13]. At this step, captured data, containing information about both the tweets and the users, are stored without any processing.

B. Filtering system

This phase is necessary because not all extracted tweets actually contain information related to malware or virus diffusion. The filtering system is based a pre-processing routine and on Naïve Bayes classifier and permits eliminating tweets containing erroneous information. In particular, every tweet is pre-processed in order to delete all words that are not significant for classification, i.e., punctuation marks, grammatical conjunctions, some social network words (*retweet* and *@name*) and all links contained in the tweet. In information retrieval, such words that do not add relevant information are usually referred as *stopwords* [14].

Finally, in order to discard irrelevant messages which contain the chosen keywords, although without any connection with computer security, the pre-processed tweets are filtered out by using a Bayes classifier trained on a set of tweets containing an equal number of i) events related to security attacks, viruses, malware, and ii) generic messages. Generally, Bayesian systems have proved to be extremely effective in managing information with a negligible noise, even in areas where it is necessary to merge information from multiple sources [15], [16].

C. Data Analysis

The extracted and filtered data are the basis of the malware detection system. In the analysis phase, the number of tweets represents one of the parameters used to determine the threshold per unit of time, beyond which an alert should be reported. Unfortunately, the amount of information about new computer attacks is variable, so it is not possible to accurately calculate the minimum number of tweets that must be passed to be sure a new event occurred.

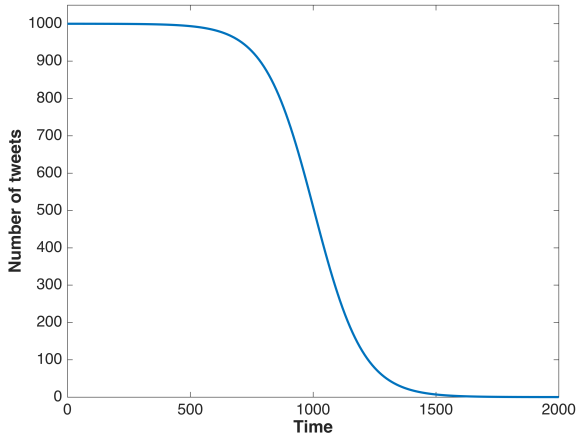

Fig. 2. Sigmoid function.

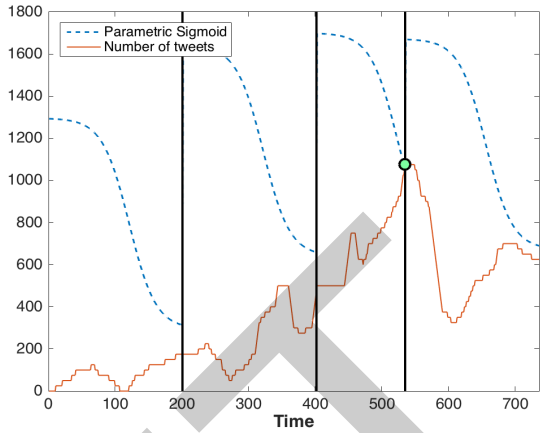

Fig. 3. Example of variable-length windows.

Fig. 4. Alert sub-system.

In the analysis phase, detection windows are dynamically terminated according to a sigmoid function, or when the maximum window size is reached.

The sigmoid function (see Fig. 2) is defined as:

$$S(x) = c1\left(1 - \frac{1}{1 + e^{-c2(x-c4)}}\right) + c3 \quad (1)$$

where *c1* is the extension of y-axis; *c2* is the asymptote of the function, *c3* is the minimum value of the function, and c4 is the central point of the abscissas. The parameters of the sigmoid function depend on *variance* and *average* values computed in each time window.

Thanks to these static values, it is possible to trace variations in the number of tweets and, so, define a minimum threshold value in order to alert a new event. This threshold is defined according to what described in [3]. Specifically, if *X[t]* represents the number of tweets recognized in interval *t*, then the *Exponential Weighted Mobile Average* (EWMA) is defined as follows:

$$Y[t] = qX[t] + (1-q)Y[t-1] \quad (2)$$

where q is chosen in a heuristic way. Subsequently, the differential value is computed as:

$$D[t] = X[t] - Y[t-1] \quad (3)$$

Through *D[t]*, the exponential mobile variance is obtained as:

$$V[t]^2 = pD[t]^2 + (1-p)V[t-1]^2 \quad (4)$$

where, also p is chosen in a heuristic way. Finally, the threshold value *T[t]* is:

$$T[t] = Y[t-1] + dV[t-1] \quad (5)$$

where d determines how much *D[t]* differ from the normal behavior *Y[t]*.

We use *T[t]* as the minimum value to exceed to alert a new event. According to the sigmoid function showed in (1), *c3* is set equal to *T[t]*, while *c1* is experimentally set to *0,3 * T[t]*. In this way the parameters of the sigmoid are configured dynamically and allowing the detection of new events when the number of tweets quickly increases. This also allows creating an alert even within the observation window, i.e., before the window has expired.

An example of variable-length windows is showed in Fig. 3. In particular, blue curves represent the parametric sigmoid functions depending on the threshold value *T[t]*, whilst the amount of tweets captured at a given time is depicted in orange. When the number of tweets exceeds the sigmoid function, the time window is closed instantaneously and an event is alerted. In the example, this moment is represented by a green point. After the event is alerted, the system starts again to analyze Twitter stream in order to find new events.

D. Alert sub-system

When an event is created, the alert sub-system (see Fig. 4) firstly groups the tweets that are related to the same topic. This phase is called document clustering and its purpose is to get homogeneous sets of posts so that it is possible identify the main topic that originated the event. To this end, each tweet is represented as a vector of terms using the Term Frequency Inverse Document Frequency (TF-IDF) metric, that associates more weight to terms that are more frequent in tweets but are rare in all other posts. In particular, this metric is defined as follows:

$$TF - IDF_{x,y} = \left(\frac{N_{x,y}}{N_{t,y}}\right) log\left(\frac{T}{T_x}\right) \quad (6)$$

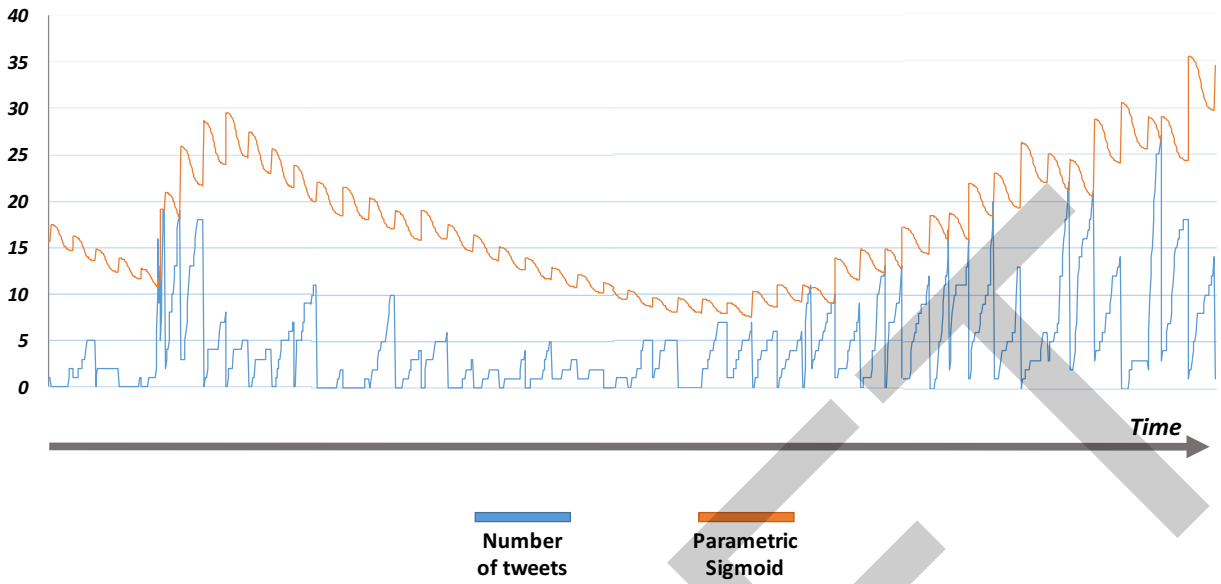

Fig. 5. Dynamic thresholding combining *Exponential Weighted Mobile Average* (EWMA) and sigmoid function. The parametric sigmoid adapts its behavior to the number of incoming tweets. An alert is created when the number of tweets exceeds the dynamic threshold.

where $N_{x,y}$ is the number of times that word *P* appears in tweet T_y, $N_{t,y}$ is the total number of words in the tweet, *T* is the total number of documents and T_x is the number of documents where word *P* appears at least once.

Subsequently, each vector of terms is used as input of a clustering algorithm, i.e., the K-Means algorithm, that will aggregate similar tweet providing also a representation of the tweet collection through the cluster centroids. Several distance functions and similarity measures can be used in document clustering, e.g., relative entropy, squared Euclidean distance, cosine similarity, and so on [17].

In our system we used cosine similarity metric, that assumes values in $[0, 1]$, and is defined as follows:

$$SIMcos(\overrightarrow{d_1}\overrightarrow{d_2}) = \frac{\overrightarrow{d_1} \cdot \overrightarrow{d_2}}{|\overrightarrow{d_1}||\overrightarrow{d_2}|} \quad (7)$$

where d_1 and d_2 represent n-dimensional vectors over a set of terms $A = \{a_1, \ldots, a_m\}$.

Since we want to create a text alert to be shared on Twitter, a summarization [18] step is performed so that the particular event can be uniquely represented by a short text. To this aim, for each event/cluster, tweets are compared with the cluster barycenter and the tweet that is closest to the cluster's center is selected as representative of the whole cluster.

E. Feedback analysis

A very important element of the architecture we propose is the analysis of the feedback provided by the users on the quality of the alert. This information is used as input of a Bayes classifier to start a refinement procedure aimed at improving the overall system performance.

In order to determine how users' feedback can influence the classification system, some metrics have been defined to measure whether a Twitter user is influential or not, and to evaluate his reports as "good" or "bad". In other words, the system is capable of training Bayes classifier only if users who have left feedback through the usual mechanisms made available by Twitter, as "I like" and "Retweet", are influential users. Influence can be described as the ability of a person to influence the behavior and actions of another person. The parameter used to define an influential user depends on two factors: i) the *InterestRate*, indicating the number of active contributions a user received on his posts, over ii) the user *affiliation*, that capture the relation between the number of user's friends and followers, i.e., influential users have generally more followers than friends which results in a lower *affiliation* value.

IV. Experiments

As discussed so far, the system we propose is able to detect an event when the number of tweets related to a new malware event exceeds the dynamic threshold. Then, the collected tweets are processed and summarized to create an alert. Finally, the users can provide their feedbacks on the reported events in the form of likes or retweets.

Experimental analysis focused on the evaluation of the entire system in a real scenario. The proposed architecture has been used to capture from the Twitter stream a set of tweets

TABLE I
EVENTS DETECTED BY THE PROPOSED SYSTEM

US Government Employees Targeted By New GovRAT Malware
3 cybersecurity posts federalbureauofinvestigation malware
Seagate Central NAS boxes hit Miner-C malware
New malware makes money “borrowing” computer mine cryptocurrency
Watch nasty Macbook malware” applenewsfeedly
Android Banking Apps Targeted By New Malware android Apps
Macro-based malware Hancitor developed ways trick user
Hackers hit Seagate NAS devices cryptomining malware
New post: Malware Alert! Increasing Threats Put Bitcoin Users Danger
New DressCode malware slips Google Play Store undetected
Researchers warn hackers DDoS 911 emergency phone service infosec
Hackers hit Seagate NAS devices cryptomining malware

containing some relevant keywords, i.e., “spam”, “malware”, “worms”, “CyberSecurity”, “infosec” and others.

The tweet collection has been processed within time windows in order to promptly detect new events, create a report about them, and share the alert on Twitter itself. Fig. 5 shows the number of tweets (blue) captured during different windows (x-axis) and the corresponding values of the threshold, computed in the same period, represented by parametric sigmoid function (orange). The curves show different phases of the alerting system, both when the threshold value is higher than the number of tweets, and when such number exceeds the threshold causing an alert. Thanks to the dynamic thresholding mechanism, the parametric sigmoid follows the trend of statistical data; thus, if the number of tweets decreases, then also the detection threshold gradually decreases.

Of particular interest are the moments in which the number of tweets exceeds the parametric sigmoid. When this happens, the system increments the threshold value so that it can be used to find other upcoming events. Results reported in TABLE I addresses the period analyzed in Fig.5, where 12 events regarding new cyber-attacks and malware threats have been detected and reported.

Other experiments were carried out to evaluate the user’s feedback mechanism.

As discussed in the previous paragraphs, users can give feedbacks on the alerts reported by the system. Such feedbacks to improve the detection mechanism according to the user’s degree of influence. In order to compute the *threshold of influence*, i.e., the value that determines whether a user is influential and his feedback is useful or not, experiments were performed on a sample of 24 different users focusing on the number of friends (*FRN*) and followers (*FLW*), *affiliation* and *interest rate*. Specifically, these metrics are calculated as:

$$affiliation = \frac{FLW(ids) \cap FRN(ids)}{FLW(ids) \cup FRN(ids)} \tag{8}$$

$$\begin{aligned} intRate = Conversation + Favorites + \\ + Mentions + Replies + Retweet \end{aligned} \tag{9}$$

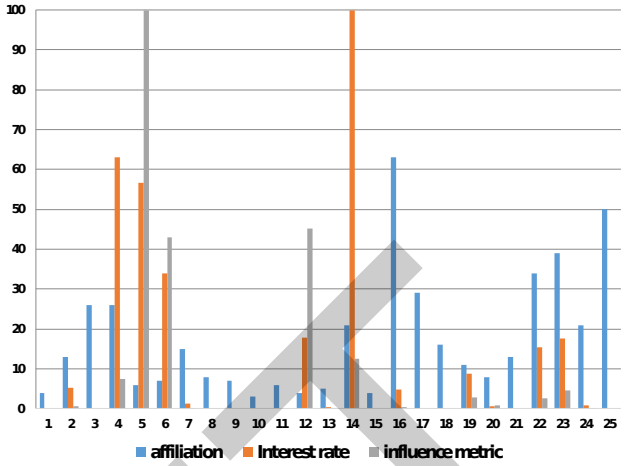

Fig. 6. User influence analysis.

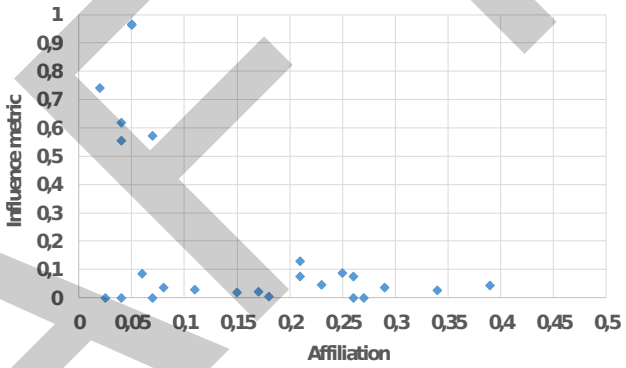

Fig. 7. Relationship between influence and affiliation metrics.

$$influence\_metric = \frac{intRate}{affiliation(\frac{FRN}{FLW})} \tag{10}$$

The most important parameter is the *influence metric*, which represents the degree of interaction between a user and the community by means of the Twitter functionalities (Retweet, feelings, answers, like). When this metric assumes high values, then the probability that the user is influential is very high.

First, we analyze the user influence according to *affiliation*, *interest rate* and *influence metrics*. The results obtained from this first phase of analysis (see Fig. 6) showed that the most influential users have a low level of *affiliation* since they tend to be followed by a huge number of users. Generally, *affiliation* is a metric that assumes values between 0 (no matches) and 1, when the set of friends and followers coincide. Thus, we can conclude that, for influential users, the two sets of followers and friends are almost completely disjointed.

In order to establish the value that marks a user as “influential” and, then, trustworthy, it is necessary to analyze the relationship between the influence metric and the *affiliation* values. To this end, we plotted such relationship (see Fig. 7) where x-axis represents *affiliation* values and y-axis shows the influence metric normalized in the range $[0, 1]$. Results confirm that users with a high degree of *affiliation* have very low

influence, and higher values of *influence metric* are obtained when the *affiliation* assumes values lower than 0,1. According to these results, it is possible to define a threshold value of the *influence metric* beyond which the system can consider a user as influential. In particular, a user can be considered as "influential" if his *influence metric* value exceeds the value 0.5.

V. Conclusion

In this paper we presented an intelligent to system aimed at analyzing the Twitter stream in order to create automatic alerts when news about malware attacks, or other computer security threats, spread through the Web.

The key components of the architecture we propose are the use of dynamic-length observation windows and dynamic threshold, which let the system adapt its behavior to the volume of tweets captured at a given time. Moreover, the adoption of some pre-processing steps and a Naïve Bayes classifier helps to improve the performances of the alerting system by filtering out irrelevant tweets.

Experiments have been performed to validate the effectiveness of the proposed solution and as a result some intrusive and exemplary malware activities currently circulating on the network have been detected.

As future work we want to extend the system functionalities to provide alerts on generic events according to the any set of initial keywords. Moreover, the set of keywords used to extract information from the Twitter streaming should be automatically expanded in order to increase the chance of detecting significant events.

Finally, we want to investigate the adoption of more sophisticated techniques to estimate the trustworthy of the users and, consequently of their feedbacks. To this aim, we plan to provide the system with a reputation management algorithm, such as those reported in [19], [20], in order to ensure the robustness of reputable systems against attacks on their security [21], [22].

References

[1] Y. Erkal, M. Sezgin, and S. Gunduz, "A new cyber security alert system for twitter," in *Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on*. IEEE, 2015, pp. 766–770.

[2] S. Gaglio, G. L. Re, and M. Morana, "A framework for real-time twitter data analysis," *Computer Communications*, vol. 73, pp. 236–242, 2016.

[3] I. Al-Qasem, S. Al-Qasem, and A. T. Al-Hammouri, "Leveraging online social networks for a real-time malware alerting system," in *Local Computer Networks (LCN), 2013 IEEE 38th Conference on*. IEEE, 2013, pp. 272–275.

[4] S. Abraham and I. Chengalur-Smith, "An overview of social engineering malware: Trends, tactics, and implications," *Technology in Society*, vol. 32, no. 3, pp. 183–196, 2010.

[5] G. Yan, G. Chen, S. Eidenbenz, and N. Li, "Malware propagation in online social networks: nature, dynamics, and defense implications," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*. ACM, 2011, pp. 196–206.

[6] W. Xu, F. Zhang, and S. Zhu, "Toward worm detection in online social networks," in *Proceedings of the 26th Annual Computer Security Applications Conference*. ACM, 2010, pp. 11–20.

[7] M. Xie, Z. Wu, and H. Wang, "Honeyim: Fast detection and suppression of instant messaging malware in enterprise-like networks," in *Computer Security Applications Conference, 2007. ACSAC 2007. Twenty-Third Annual*. IEEE, 2007, pp. 64–73.

[8] F. Atefeh and W. Khreich, "A survey of techniques for event detection in twitter," *Computational Intelligence*, vol. 31, no. 1, pp. 132–164, 2015.

[9] M. Cordeiro, "Twitter event detection: combining wavelet analysis and topic inference summarization," in *Doctoral symposium on informatics engineering*, 2012, pp. 11–16.

[10] P. Rao, C. Kamhoua, L. Njilla, and K. Kwiat, "Methods to detect cyberthreats on twitter," in *Symposium on Security and Privacy in Social Networks and Big Data (SocialSec)*, 2016, p. 33.

[11] A. L. Hughes and L. Palen, "Twitter adoption and use in mass convergence and emergency events," *International Journal of Emergency Management*, vol. 6, no. 3-4, pp. 248–260, 2009.

[12] S. Lee and J. Kim, "Warningbird: A near real-time detection system for suspicious urls in twitter stream," *IEEE transactions on dependable and secure computing*, vol. 10, no. 3, pp. 183–195, 2013.

[13] S. Gaglio, G. L. Re, and M. Morana, "Real-time detection of twitter social events from the user's perspective," in *Communications (ICC), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1207–1212.

[14] A. Singhal, "Modern information retrieval: A brief overview," *IEEE Data Eng. Bull.*, vol. 24, no. 4, pp. 35–43, 2001.

[15] A. De Paola, S. Gaglio, G. L. Re, and M. Ortolani, "Multi-sensor fusion through adaptive bayesian networks." in *AI* IA*. Springer, 2011, pp. 360–371.

[16] A. De Paola, M. La Cascia, G. L. Re, M. Morana, and M. Ortolani, "User detection through multi-sensor fusion in an ami scenario," in *Information Fusion (FUSION), 2012 15th International Conference on*. IEEE, 2012, pp. 2502–2509.

[17] A. Huang, "Similarity measures for text document clustering," in *Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand*, 2008, pp. 49–56.

[18] K. Rudra, S. Ghosh, N. Ganguly, P. Goyal, and S. Ghosh, "Extracting situational information from microblogs during disaster events: a classification-summarization approach," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015, pp. 583–592.

[19] V. Agate, A. De Paola, G. L. Re, and M. Morana, "A simulation framework for evaluating distributed reputation management systems," in *Distributed Computing and Artificial Intelligence, 13th International Conference*. Springer, 2016, pp. 247–254.

[20] C. Crapanzano, F. Milazzo, A. De Paola, and G. L. Re, "Reputation management for distributed service-oriented architectures," in *Self-Adaptive and Self-Organizing Systems Workshop (SASOW), 2010 Fourth IEEE International Conference on*. IEEE, 2010, pp. 160–165.

[21] V. Agate, A. De Paola, S. Gaglio, G. Lo Re, and M. Morana, "A framework for parallel assessment of reputation management systems," in *Proceedings of the 17th International Conference on Computer Systems and Technologies 2016*. ACM, 2016, pp. 121–128.

[22] V. Agate, A. De Paola, G. L. Re, and M. Morana, "Vulnerability evaluation of distributed reputation management systems," in *proceedings of the 10th EAI International Conference on Performance Evaluation Methodologies and Tools on 10th EAI International Conference on Performance Evaluation Methodologies and Tools*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2017, pp. 235–242.