OSINT-based Data-driven Cybersecurity Discovery

Fernando Alves
Pedro M. Ferreira (advisor)
Alysson Bessani (advisor)
falves@lasige.di.fc.ul.pt
pmferreira@fc.ul.pt
anbessani@fc.ul.pt
LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal

ABSTRACT

The cyber-arms race burdens security analysts with a constant search of timely threat and security data. Skimming through various OSINT sources is a time consuming task, for which security analysts have a limited budget. This thesis presents SYNAPSE — a framework for the collection, classification, and aggregation of tweets for security purposes. SYNAPSE is designed to be deployed in a production environment, where it will constantly manage its machine learning classifier, and actively search for adequate cybersecurity tweeters.

1 INTRODUCTION

Cybersecurity is a matter of growing concern as cyberattacks cause loss of income, sensitive information leaks, and even vital infrastructures to fail. To streamline the security management of organizations with a wide IT infrastructure, *Security Information and Event Management* (SIEM) systems [8] can be employed. SIEMs aggregate event data produced by security devices, network infrastructure, systems and applications. As with any security system, SIEMs must be constantly updated with the latest security content to maximize their threat coverage and detection. Although the quality of the detection rules directly influences the SIEM's performance, the SIEM can only detect attack signatures present on its security database.

Usually, security systems are solely updated by the company that provides them, *i.e.*, if one buys a security software from A, it will be solely updated by A's feed. This is not the case of the SIEM, as these systems are extensible and can receive intelligence from almost any source. Companies purchase additional security feeds from specialized companies not only to increase the quality of the SIEM's database, but also to complement the detection capabilities of the *Security Operations Center* (SOC) and increase the company's security level. However, focusing on a few companies' feed is a reductive approach, since a wealth of knowledge is published daily by all kinds of security information sources, such as security analysts, researchers, and hackers.

To broaden the horizons of security intelligence, companies turn to *Open Source Intelligence* (OSINT), which in summary is information publicly available on the news and web [14]. The research community has taken interest in using OSINT for security purposes, specially Twitter. There is a set of works that search for OSINT information about an IT infrastructure [9, 11, 12, 15]. These share two common characteristics: (1) a keyword set that is used to govern the selection of tweets, thereby selecting only the potentially relevant content; then, (2) another technique is used to classify the tweets as relevant or not.

However, the keyword set used to select Twitter data is a sensitive element of these proposals, as it may filter important tweets due to its possible incompleteness. Additionally, these works focus only on the data collection and classification aspects, and do not consider an end-to-end approach focused on the end-user context, that solves the problems related to the summarization and presentation of data. The content selection capabilities of the said research works use some form of machine learning model to select what is relevant to the user. These models are used to classify OSINT, which is subject to natural changes over time, as data sources differ and writing styles change. The existing research works overlook their practical applicability, and do not discuss the model's performance overtime, nor any strategies for model maintenance. Machine learning systems are known to lose performance when classifying data different from its training set, *i.e.*, a model's performance is expected to drop overtime when classifying a changing event stream.

Although there are many sources of OSINT, Twitter was used for two main reasons. First, Twitter is well-recognized as an important source of short notices (almost in real-time) about web activity and occurring events [1]. This is true also for cybersecurity-related events, as demonstrated by the highly-active accounts of most security feeds and researchers, where they tweet security-related news [3, 7, 12]. Therefore, Twitter is an interesting aggregator of information and activity from all kinds of sources. Second, since a tweet is limited to 280 characters (mostly 40–60 words), these messages are potentially simple to process automatically, enabling very high levels of accuracy and low false positive rates through standard machine learning techniques.

2 RESEARCH QUESTIONS

This work proposes the following thesis hypothesis:

Security systems can be enhanced with suitable Open Source Intelligence using a self-maintained system.

In the following sections are presented this thesis' research questions. For each question we present an overview of how to answer it, and its evaluation methodology.

2.1 What are the advantages of using Twitter as an OSINT source?

Twitter is considered very useful as a natural aggregator of current events, including security notices, as has been noted by the research community. Although used in several research works and many

EuroDW'18, April 23rd, 2018, Porto, Portugal
2018.

OSINT collection tools include tweet collecting capabilities, there are no qualitative studies concerning the data published on Twitter. This research goal is meant to assess a set of aspects about the data published on Twitter, focused on Twitter's timeliness in disclosing threat data.

To perform the study we will use the VepRISK database [2], which contains a copy of some of the main vulnerability databases (*e.g.*, NVD, CVE). We will search for tweets mentioning the threats contained in VepRISK, and compare the dates and other data to answer the above questions. Since Twitter's API provides access to only the latest week of tweets, we will use the GetOldTweets[1] library, which is capable of retrieving tweets from any time, thus allowing us to bypass Twitter's data collection restrictions.

Evaluation. We will compare the data collected from Twitter with a set of vulnerability and exploit databases, collated in the VepRISK database [2].

2.2 How to create a framework capable of collecting and selecting only relevant OSINT for a given IT context, and summarize the collected information for the convenience of SOC operators?

The first phase of SYNAPSE's development is focused on its core functionalities: the data collection, selection, and summarization. The proposed pipeline begins with the tweet collection module. This module receives from the user a set of Twitter accounts, from which it will collect every tweet posted. The following module is a filter, that receives from the user a set of words describing the IT infrastructure where SYNAPSE will be deployed (hereafter called *infrastructure context*). The filter drops any tweet that does not contain mention any elements of the infrastructure.

The tweets who pass the filter will undergo a pre-processing phase, where stopwords are removed and the text itself is normalized (*e.g.*, convert all words to lower case). Following, we will have a feature extraction phase, where we obtain a numerical vector for each tweet using TF-IDF [17]. Now that we have a numerical format for each tweet, we can have these classified by a supervised machine learning approach. We will compare the performance of two classifiers: the SVM and the MLP ANN, both with a long track of good results over various types of classification tasks.

After being classified, the tweets are temporarily stored until δ tweets have been classified or θ time has passed (whichever comes first), at which point these will be sent to the clustering module, that uses the k-means clustering function to group them by similarity. Clustering is a fundamental step of this pipeline since it avoids presenting the same data multiple times. From each cluster is selected a tweet that represents the data present on that cluster, called the *exemplar*.

Evaluation. The framework's core will be evaluated through the classifier's TPR and TNR, and through its capability to summarize data correctly, *i.e.*, the generated cluster are different among themselves and each cluster contains only very similar elements.

2.3 How generate appropriate IoCs from the collected OSINT?

SYNAPSE will include a module to generate IoC. We need to choose an appropriate standard to generate IoCs in before beginning this phase's implementation. The OpenIoC[2] format and the MISP platform are strong candidates due to their versatility, although we will evaluate other options.

The first implementation step is to encapsulate each cluster in an IoCs, highlighting the exemplar but providing access to the other cluster elements. Then, from the exemplar we can extract some useful elements to make the IoC information richer. We can use named entity recognizers[10] to obtain the crucial elements of a tweet (attack, vector, target), similarly to what has been performed by Liao *et al.* [6], and add that information content to the IoC. Further, each IoC should categorized by type, such as attack, vulnerability, or patch. We also intend to add a priority value to each IoC, which could extend or complement similar existing metrics. If during implementation we discover that there are too many metrics, we can use fuzzy logic [5, 16] to generate a single priority value.

The final contribution will be to match the IoC's target and type with an action, such as *the <attack> on <target> can by contermeasured by performing <action>*. We will also investigate if it is possible to trigger an automatic search for patches or updates of that IT element.

Evaluation. The first phase's evaluation is to verify that the IoCs are generated correctly. The second phase is evaluated through the number of cases it can create a correct IoC. The third phase will be evaluated using TPR and TNR, by observing the matches performed.

2.4 How to build a self-managed classifier and account selector?

Twitter provides two methods to collect tweets through their API: *a)* all tweets posted by a set of user accounts, or *b)* all tweets containing one or more words from a word set. SYNAPSE will include a module to collect tweets directly from Twitter accounts; the selected accounts should mainly post security-related content. This way, the largest portion of the collected tweets should related to IT security, thus reducing irrelevant content.

Twitter accounts are being constantly added and removed, and live accounts may change their posting scope. SYNAPSE needs to adapt to these changes and strive to improve the quality of its data. The account management will be performed twofold: SYNAPSE must detect accounts that no longer post relevant content (either because the account's scope changed or the account is inactive), and must constantly search for accounts posting data relevant to its infrastructure context.

The classifier must also be managed overtime, since its decisions influence all aspects of SYNAPSE's performance, and it is crucial to maintain its quality. It has a central role in SYNAPSE, since it "decides" what data is deemed relevant, and which accounts are added and removed. Most tweets are written by people, who are expected to differ on two aspects overtime: the writing style and the nature of the posts. The classifier's performance is expected

[1] https://github.com/fernandoblalves/GetOldTweets-java/

[2] http://www.openioc.org

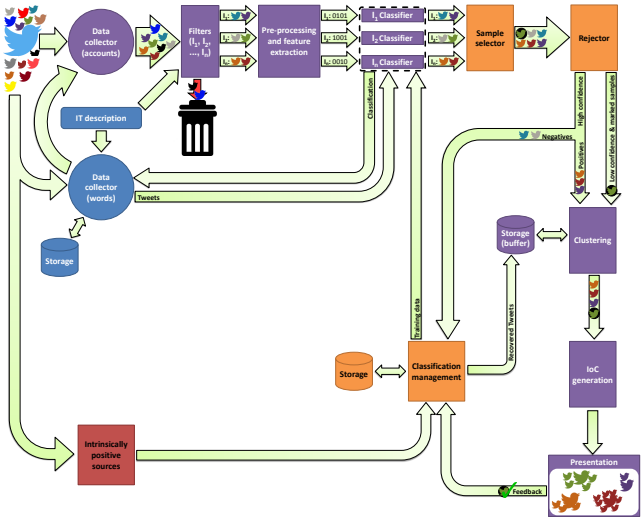

Figure 1: SYNAPSE's final architecture.

to drop overtime as it is presented with tweets different from its training set, as their content changes overtime.

The classifier will be managed through active learning [13]; SYNAPSE will include a module that observes the tweets chosen for classification, selects some of them to be validated by a human (*e.g.*, a security analyst), and uses this for classifier training. Further, to reduce the classifier's error, SYNAPSE will include rejection [4]. The classifier will provide each tweet with a label and a confidence score, representing the confidence the classifier has that the attributed label is correct. The rejection module will a select percentage of the tweets classified as negative with lowest confidence and reject them, *i.e.*, remove the attributed label and ask for manual classification. All rejected samples will be presented to the analyst, in an effort to avoid losing valuable data that was misclassified as negative.

Figure 1 presents SYNAPSE's final architecture, where the purple boxes are the core elements, the blue elements perform the maintenance of Twitter sources (although these depend on the classifier), the orange boxes represent the elements who generate feedback to manage the classifier, and finally the red box is the module that collects intrinsically positive tweets.

Evaluation. The evaluation will be performed using a new dataset currently in development, consisting of one year of classified tweets. The first month will be used to train the model, and the following to evaluate its performance in various parameters:

- Compare classifier management modes:
  - No management;
  - Incremental training;
  - New model with all labelled data;
  - New model with the latest N months of labelled data.
- Evaluate the impact of rejection on the classifier performance:
  - Does our dataset present a correlation between error and rejection?
  - Does rejection reduce significantly the FPR?

3 DISCUSSION

To train and evaluate SYNAPSE we are currently developing a tweet dataset composed of labelled tweets collected during a one year period. These tweets concern three IT infrastructures, whose description was provided by the three industrial partners of the DiSIEM project,[3] where this PhD thesis is inserted.

Without adequate maintenance, a classifier's quality is assumed to drop in the face of data distinct to its training set. The proposed methodology is expected to maintain the classifier's TPR and TNR overtime through the help of a human oracle. Further, we predict a very low FNR, which is very important in this context since missing security updates denies the advantage of employing a system such as SYNAPSE. It is possible that on the first classifier training cycles, the analyst is presented with a high volume of false positives and oracle requests. However, we expect this behaviour to reduce once the classifier's training set increases and adapts itself to the infrastructure context.

It is also possible that querying the oracle does not provide enough labelled data to train and manage the classifier. In that case, the *Intrinsically positive sources* module will have to largely compensate the lack of labelled samples, which is dangerous since this module's output will not be human verified.

Another possible issue is related to the Twitter account used to collect tweets. To use Twitter's API, one requires a developer account. Each account has a limited number of accounts it can follow (5000), and number of tweets it can receive per second when streaming — 1% of the total tweets published on that moment. In case any of these limits is reached, we will have to automatically add a new developer account and split the followed accounts by our following accounts.

Although there are many solutions and research works that employ OSINT for security purposes, there are clear gaps. Almost all research works do not provide their data in a machine readable format, nor designed their proposal taking into account the reality of a SOC. Further, these are static approaches, that do not take into account the necessary management of their systems — specially the machine learning models. In contrast commercial tools are capable of little to no processing, which is crucial for its efficient usage. Our proposed approach aims at filling all these gaps, providing a complete end-to-end solution that:

- Is selective on its data sources;
- Filters irrelevant information by design;
- Uses a machine learning classifier to infer a tweet's relevance;
- Clusters tweets to collate repeated items;
- Generates security alerts in the IoC machine readable format;
- Automatically manages its data sources;
- Automatically manages its classifier.

4 ACKNOWLEDGEMENT

This work was supported by EC through funding of DiSIEM project, ref. G.A. n°700692, and by FCT through funding of LASIGE Research Unit, ref. UID/CEC/00408/2013.

[3] http://disiem-project.eu/

REFERENCES

[1] [n. d.]. How people use Twitter in general - American Press Institute. https://www.americanpressinstitute.org/publications/reports/survey-research/how-people-use-twitter-in-general. ([n. d.]). [Accessed 12-03-2018].

[2] Ambrose Andongabo and Ilir Gashi. 2017. vepRisk - A Web Based Analysis Tool for Public Security Data. In *13th European Dependable Computing Conference, EDCC 2017, Geneva, Switzerland, September 4-8, 2017*.

[3] Rodrigo Campiolo et al. 2013. Evaluating the Utilization of Twitter Messages As a Source of Security Alerts. In *Proc. of the 28th ACM SAC*.

[4] C Chow. 1970. On optimum recognition error and reject tradeoff. *IEEE Transactions on information theory* 16, 1 (1970), 41–46.

[5] Jyh-Shing Roger Jang, Chuen-Tsai Sun, and Eiji Mizutani. 1997. Neuro-fuzzy and soft computing; a computational approach to learning and machine intelligence. (1997).

[6] Xiaojing Liao et al. 2016. Acing the IOC game: Toward automatic discovery and analysis of open-source cyber threat intelligence. In *Proc. of the 23rd ACM CCS*.

[7] Nikki McNeil et al. 2013. PACE: Pattern accurate computationally efficient bootstrapping for timely discovery of cyber-security concepts. In *Proc. of the 12th ICMLA*.

[8] David R Miller, Shon Harris, Allen Harper, Stephen VanDyke, and Chris Blask. 2010. *Security Information and Event Management (SIEM) Implementation (Network Pro Library)*. McGraw Hill.

[9] Sudip Mittal et al. 2016. Cybertwitter: Using twitter to generate alerts for cybersecurity threats and vulnerabilities. In *Proc. of the 8th IEEE/ACM ASONAM*.

[10] Behrang Mohit. 2014. Named entity recognition. In *Natural language processing of semitic languages*. Springer, 221–245.

[11] Alan Ritter et al. 2015. Weakly supervised extraction of computer security events from twitter. In *Proc. of the 24th WWW*.

[12] Carl Sabottke et al. 2015. Vulnerability disclosure in the age of social media: exploiting twitter for predicting real-world exploits. In *Proc. of the 24th USENIX Security Symp*.

[13] Burr Settles. 2012. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 6, 1 (2012), 1–114.

[14] Robert David Steele. 1996. Open source intelligence: What is it? why is it important to the military. *American Intelligence Journal* 17, 1 (1996).

[15] Robert Tibshirani et al. 2001. Estimating the number of clusters in a data set via the gap statistic. *J. Royal Stat. Soc., Series B* 63, 2 (2001).

[16] Lotfi A Zadeh. 1996. Fuzzy sets. In *Fuzzy Sets, Fuzzy Logic, And Fuzzy Systems: Selected Papers by Lotfi A Zadeh*. World Scientific, 394–432.

[17] Mohammed J Zaki et al. 2014. *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press.