Cyber Semantic Web

Dipesh S. Patel
School of Computer Science
University of Windsor
Windsor, Canada
patel21s@uwindsor.ca

Parth A. Shukla
School of Computer Science
University of Windsor
Windsor, Canada
shukl11g@uwindsor.ca

Manan H. Patel
School of Computer Science
University of Windsor
Windsor, Canada
patel1wz@uwindsor.ca

***Abstract*—In todays era, microblogging platforms serve as the source for a plethora of sensitive and diverse information. We describe Cyber Semantic Web a system to discover and analyze cybersecurity intelligence on Twitter and serve as an intelligence source. Real-time information in the form of tweets is extracted about various cyber threats. The intelligence gathered will be represented by the semantic web and then extracted to be used for security analysis. In this implementation, we extract a vast amount of raw data from Twitter. After extraction of data, this data is preprocessed to a form that consists of only the tweet, the data, and the URL. The preprocessed data is then stored within a processed data store. The processed data store is then converted to a cross-platform independent data store that is fed to the Open Calais API created by Thomson Reuters. With the help of the Open Calais API, a series of categories are generated with the help of Natural Language Processing (NLP). We have now obtained data with context. This context can help inform users on a specific cybersecurity category in which a tweet may best fit. This procedure has proven to be a successful implementation which allows us to input a cybersecurity category, and a series of tweets which fall under the inputted category are retrieved. These retrieved tweets can now serve as tools for cyber security analysis and as a daily insight into the world of cybersecurity.**

***Index Terms*—Natural language processing, Semantic Web, Cybersecurity, Contextual Information, Twitter**

I. Introduction

In a domain as vast as cybersecurity, professionals require knowledge that is an accurate representation of the state-of-the-art technology that is currently being used. This allows to make critical and strategic decisions. The knowledge must be extracted from a variety of independent sources, and then represented in a way that is usable for further analysis or decision making. This knowledge can come from social networking sites, newspapers, blogs, etc. To extract this data manually becomes an extremely tedious task. Automatic extraction of this data has proven to be a viable research topic. We have chosen Twitter as our primary source of knowledge. The real time nature of the information can provide a significant amount of comprehension during peak influential situations. Companies such as GitHub, Kaspersky, and The Hacker News report on certain security incidents via Twitter. Independent ethical hackers also use the platform to expose potential security flaws in software and hardware systems. Therefore, we believe Twitter is a respectable source to gather information about cybersecurity threats. In our system, we first gather the information via Twitter. After preprocessing the tweets, we identify, tag, and extract insights related to cybersecurity vulnerabilities such as hardware affected, consequences of the attack, vendors, etc. with the help of the Open Calais API. Cyber Semantic Web takes the information provides and performs natural language processing to generate the most important categories of cybersecurity alerts for review. Due to the unreliable nature of Twitter data, human interaction is recommended with the system. In this experiment, our data is actionable information which can provide insights about new threats or vulnerabilities in software or hardware systems. The rest of this report is organized as follows We discuss the related work in Section 2, our framework in Section 3. Our execution and evaluation are explained in Section 4. Our discussion on the results are provided in Section 5 and our conclusion is provided in Section 6. The future work will be presented in section 7 of the report.

II. Related Works

A. CyberTwitter

In this research paper, authors discover a system to analyze cybersecurity intelligence on twitter and serve as an open source intelligence (OSINT). Semantic web resource description framework to represent the gathered information and SWRL rules to reason over and generate security alerts. A lot of information is published on cybersecurity within two main sources, formal and informal. Twitter, Reddit, etc. comes under informal sources [1]. Whereas NVD (NIST Vulnerability database), US-CERT comes under formal sources. After the data is obtained, it is passed through the security vulnerability concept extractor where its tagged [1]. Then it will link to existing concepts and entities external, publicly available semantic knowledge to enhance their extracted data. After developing ontology, they used it along with SWRL rules to address time sensitivity nature of cybersecurity events as the nature of intelligence in any security system has a temporary dimension, i.e. one information which is very useful right now might be not important after sometimes[1]. They use unified Cybersecurity ontology to provide domain information to the system, RDF linked data is stored which allows the alert system to reason over the data. Finally, alerts are issued to the end-user based on their profiles. From the user system profile, they can determine which user is running the specific version of particular operating system, so threats can be predicted upon that [1]. For the collected twitter data, they limit for English language only, then by using SVCE, it tags every concept like means, consequences of attack, affected hardware and software etc. In this system they only keep those tweets which contains two or more tags generated by SVCE [1]. The only disadvantage is that this uses open source intelligence, some might misuse the available platform to misguide the security analysts.

B. Finding Malicious Cyber Discussions in Social Media

As cyber discussions are done socially by stack exchange, Reddit, Twitter etc., it is convenient to apply natural language machine learning techniques over those datasets. As the anticipation of security analyst enhances, the organization is more likely to recover faster from cyber-attacks or steps to prevent them can be taken. Researchers have trained their classifiers using 3 social media forums: stack exchange, Reddit and Twitter because these corporation contains text with cyber content and past history can be derived for a reasonable time span [2]. For stack exchange, tags such as penetration test, buffer overflow, denial of service (DOS attack) are gathered while for Reddit, all posts under cyber topics like reverse engineering, security, malware were collected. For twitter data, 127 cyber experts were identified under the label cyber, while other random 500 users labeled non-cyber. Pipeline is required as it is unstructured data. Term frequency and inverse document frequency (TF-IDF) features were used, generated by counting the occurrence of words and number of documents in which words occur [2]. Researchers found that logistic regression classifiers and support vector machine classifier gives similar performance. Results were generated using 10-fold cross validation performed separately on each corpus. False alarm can be labeled (if non-cyber classified as cyber) or misses (cyber classified as non-cyber). If assumption is made that 5 percent cyber documents are preselected which has 1 percent false and 10 percent misses will give 83 percent cyber document filtered stream, means analyst needs 16.6 times fewer non-cyber documents to find cyber documents [2]. Classifier gives accurate results even after the discussion occurring six months or year after training. A process of classifier can be made even with 5 percent cyber documents into 83 percent and 95 percent cyber documents [2]. Lack of cross domain performance, i.e. performance diminishes when the classifier tested on one while trained on another corpus [2]. No automatic link was provided between an attacker and victim based on the data.

III. Approach

We have developed Cyber Semantic Web as a system that automatically detects cybersecurity vulnerabilities from relevant tweets that have been gathered through querying the Twitter API. The tweet gathering component consists of the collecting, cleaning, and storage of tweets into a data store. The format of the tweets stored are in the form of date, tweet, and URL. These tweets are then fed into the tagger and various tags are generated using natural language processing. After all tags are generated, the tags and their respective tweets are stored in an independent data store that is linked to the front end of the system. When a user enters a specific cybersecurity category all tweets that fall within that category will be retrieved for the user. The user may use this retrieved data to gain further insight or for analytical purposes.

A. Implementation

In this implementation we collect the data with the help of the Twitter Stream API. The data is extracted based on the user accounts that have been provided. These accounts have manually been chosen based on the quality of tweets that are provided in real-time. One limitation is that all the tweets that have been chosen are only in the English language. As you

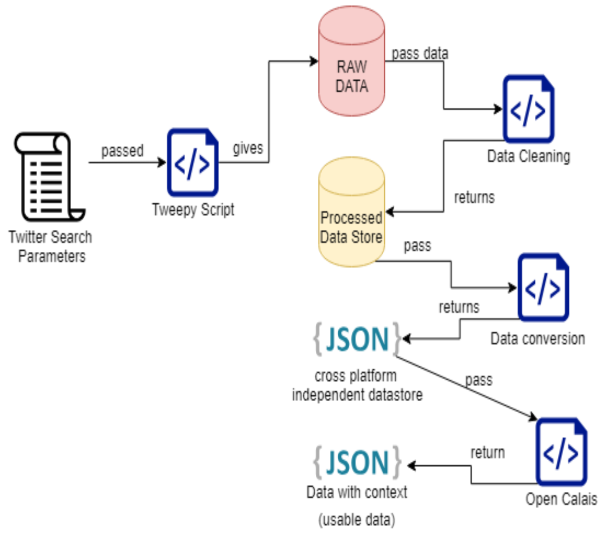


Fig. 1. Cyber Semantic Web: A framework to monitor and analyze cybersecurity threats

can see in figure 1, Twitter search parameters are not only dependent on user accounts, we can also retrieve tweets on the basis of keywords, up-and-coming topics, and hashtags. The Tweepy Script allows us to retrieve the data by querying the Twitter data store based on the parameters that have been specified in the Twitter search parameters. The output of this is raw data that is stored in a primary data store. The data in the primary data store is raw because of the inconsistency in the data format which consists of: emojis, encoding issues, redundant data, and retweet labels. To perform analysis on the data we must discard the data for efficient processing. This was done by the data cleaning script. The cleaned data is then stored into a processed data store which can then be used for further analysis and querying. When we continuously query a data store, it becomes computationally expensive and it comes with platform dependencies. So, to make things simpler we developed a script that converted the traditional data store to a custom formatted JSON file. The format of each tweet can be seen in figure 2. This is only one sample tweet out of a plethora of tweets. Our data store is inundated with such tweets.

Initially we used Protege [3] to implement our very own semantic web ontology to derive relevant insights, but to

```
{
    "tweet_id_77":
    {
        "tweet": "Hackers in China are using Datper Trojan '",
        "date": "2018-10-21 01:09:03",
        "URL": "https://t.co/dHOCQr0rp5",
    }
}
```

Fig. 2. Tweet format in JSON

implement a knowledge base from scratch proved to be time-consuming and impractical. This is because we need to add each entry explicitly into the Protege file. Due to the vastness of the domain, some information may be unwillingly left out which would lead to an incomplete knowledge base. Our visual notation of OWL ontology developed in Protege is as shown in figure 3. We shifted to Open Calais due to above stated issue.

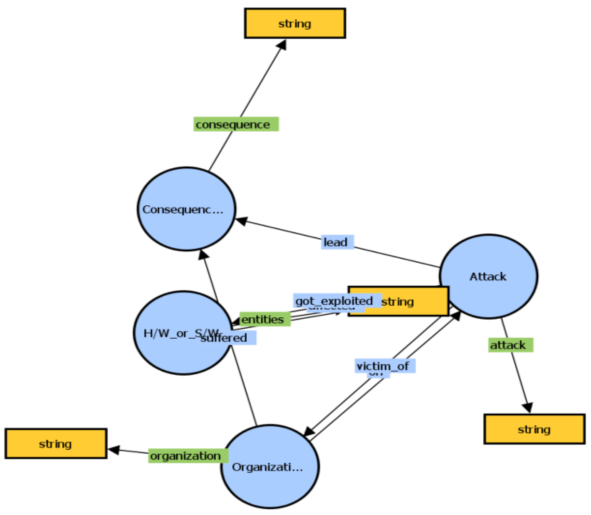


Fig. 3. Visual notation of OWL ontology

Open Calais is a software tool created by Thomson Reuters to automatically extract semantic information from the input provided. Open Calais essentially does intelligent tagging for the provided text. Intelligent Tagging gives you the quickest, simplest, and most precise approach to label the general population, spots, realities, and occasions in your substance to build its esteem, openness, and interoperability. Every substance extricated gets a significance score that precisely shows how vital that element is to the tweet. In doing this, it uncovers patterns, examples, holes, and openings that are separated for pertinence and mapped to the correct associations, subsequently making your substance more extravagant, and transforming enormous information into an increasingly exact favorable position. As discussed before the JSON data store which consists of processed tweets is then passed to the Open Calais script. This will add semantic and contextual information to the existing processed data store. The end result can be seen in figure 4 below. The Topics tag indicates

```
{
    "tweet_id_77":
    {
        "tweet": "Hackers in China are using Datper Trojan '",
        "date": "2018-10-21 01:09:03",
        "URL": "https://t.co/dHOCQr0rp5",
        "Topics": ["Techonology_Internet", "Cyberwarfare"],
        "Score": ["0.996", "0.971"],
        "Social  Tags":  ["Security",  "Malware",  "Cyberwarfare",  "Security
breaches", "Trojan horses", "Social engineering", "Spyware", "Security hacker",
"Backdoor Shell", "File binder"],
        "Importance": ["1", "1", "1", "2", "2", "2", "2", "2", "2", "2"]
    }
}
```

Fig. 4. Intelligently Tagged Tweet

the relevant topics for the tweet. The Score indicates the confidence value of the Topics that have been produced. Social Tags are the relevant key words that fits with the context of the tweet. Social Tags defines the relationship between the tweet and other relevant topics of a specific domain. Importance is the rank value of the given Social Tags. Now this JSON data store can be used for multiple uses throughout a variety of platforms as a query search engine, knowledge base, data repository, etc.

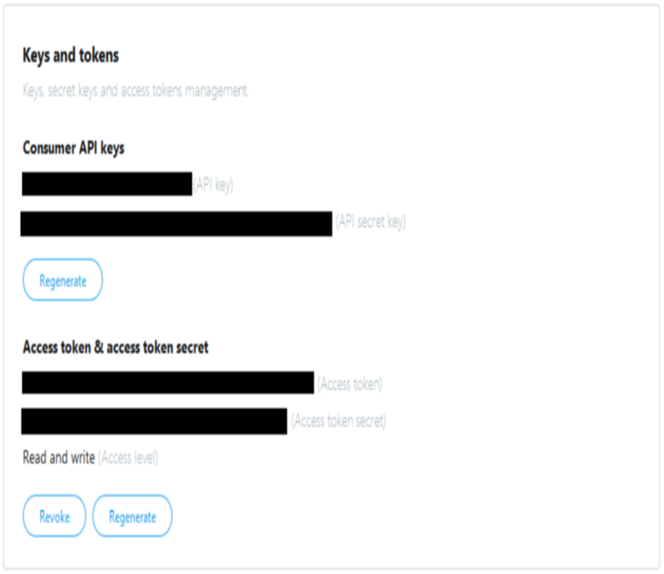
IV. Experimental Setup and Demonstration

A. Twitter API Setup

As you can in the below figure 5, these are the API keys that we used to query the Twitter database. To obtain these keys, you must register on the official Twitter developer website. After reviewing the submitted application, the Twitter team will determine whether to grant you access or not. The entire process may take anywhere from 3 to 15 business days.

B. Open Calais API Setup

To obtain the API key, you can go to the Thomson Reuters Open Calais website [4]. Once you have reached the website, you can register for the API key. After the registration is complete, the API key will be sent to the registered email address.



Keys and tokens

Keys, secret keys and access tokens management.

Consumer API keys

[redacted] (API key)

[redacted] (API secret key)

Regenerate

Access token & access token secret

[redacted] (Access token)

[redacted] (Access token secret)

Read and write (Access level)

Revoke Regenerate

Fig. 5. Twitter API Keys

C. System Calls

In figure 6 we have shown the overall flow of system calls for our framework. The main.py file consists of all the function calls to initiate the initial processes of the system. The first system call is to the tweepy.py file [5]. The tweepy.py file takes the keywords or the user accounts as search parameters from the Keywords.py file. The raw tweets extracted with the help of the Twitter stream API are then stored in a data store. The next system call is to the preprocessing.py file where all the raw tweets are cleaned and then stored in a processed data store. All the tweets in the processed data store are then converted to JSON format to ease the entire querying process when the user will enter the query. This JSON file is then passed through the context.py script where the Open Calais API will be called. In this execution, the insight that is generated will be appended into the existing JSON file. The keys appended will be the Topics, Social Tags, Score, and Importance. This will end the loop of system calls in our implementation.The assumptions in our system are:

- The main assumption of our system is that all the data gathered from the twitter, mainly the tweets from the giant organizations like GitHub, Kaspersky, and The Hacker News report are legitimate sources.
- Other assumption would be the contextual information will work better rather than just cosmetic results for a keyword.
- The ontology created is working perfectly and it is the most appropriate for our system.
- All tweets that will be extracted will be in the English language and that any emojis present within the tweet will not contribute to the insight that needs to be generated.

The development tools used in the implementation are as follows:

- Development Environment: Python 3.7
- IDE: PyCharm
- Tweepy API to get data from Twitter[5]
- Open Calais API for Natural Language Processing[4]
- Front End: HTML, CSS, Javascript (Node.js)

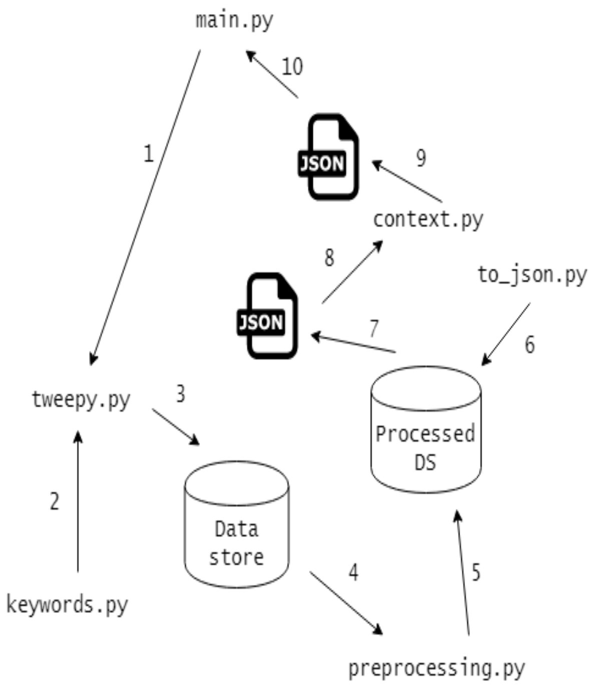


Fig. 6. System Calls

D. Activity Diagram

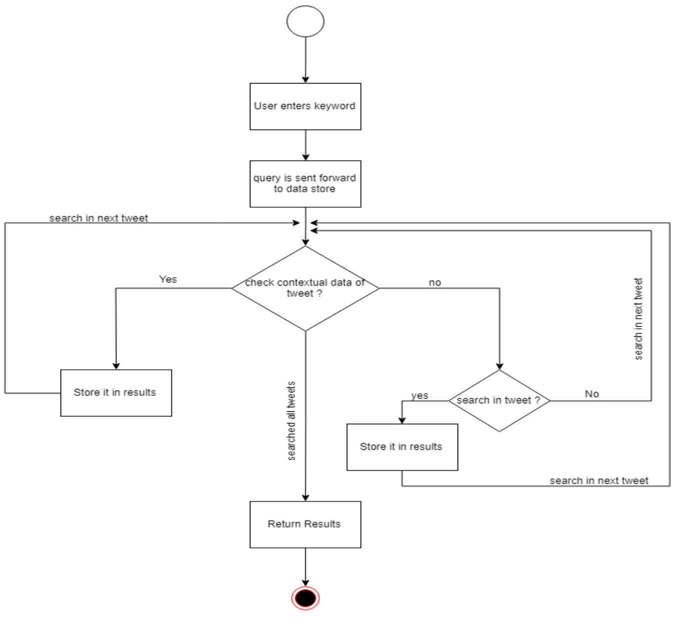
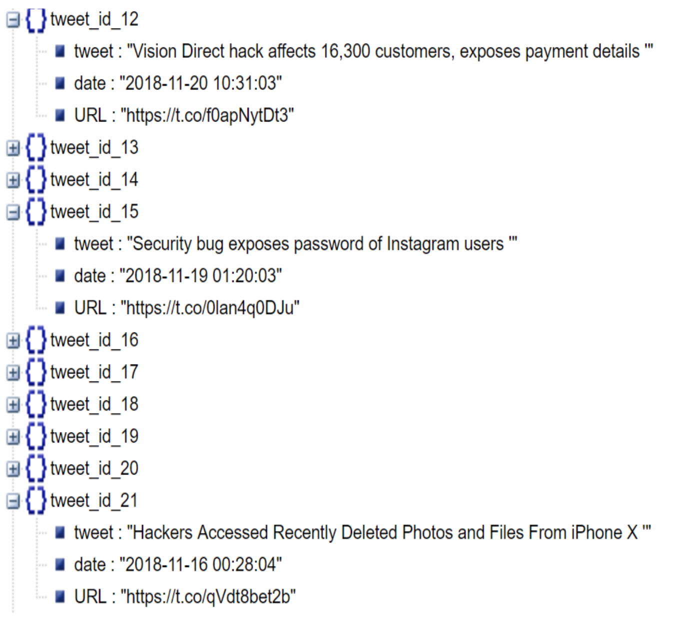


Fig. 7. Activity Diagram of System

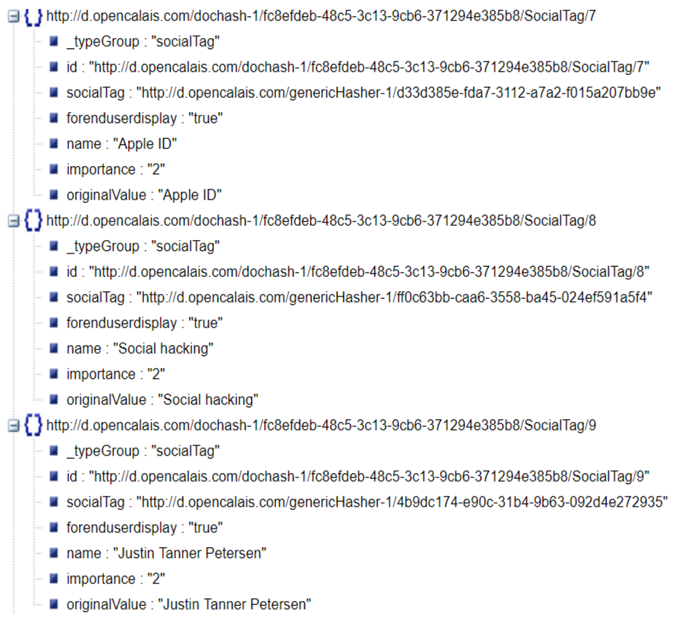
In figure 7, we can see the dynamic aspects of the system, the overall flow from one activity to another activity. The user can enter a keyword or phrase to search. This query will then be sent forward to the data store, it will now check the context of the tweet. If the context matches, the tweet will be fetched and displayed. If the context does not match the social tags, it will search within the tweet itself, and if there is a match then the tweet will be displayed to the user. Otherwise it will iteratively traverse through all the tweets. If no context is matched, the data will not be retrieved.

E. Results



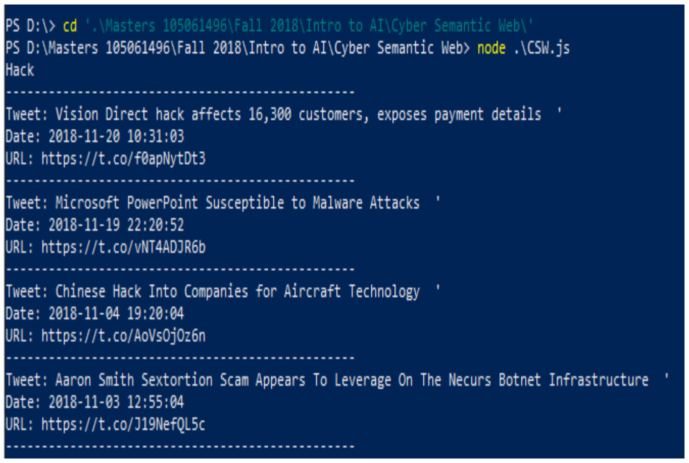
```
{ } tweet_id_12
    tweet : "Vision Direct hack affects 16,300 customers, exposes payment details '"
    date : "2018-11-20 10:31:03"
    URL : "https://t.co/f0apNytDt3"
{ } tweet_id_13
{ } tweet_id_14
{ } tweet_id_15
    tweet : "Security bug exposes password of Instagram users '"
    date : "2018-11-19 01:20:03"
    URL : "https://t.co/0lan4q0DJu"
{ } tweet_id_16
{ } tweet_id_17
{ } tweet_id_18
{ } tweet_id_19
{ } tweet_id_20
{ } tweet_id_21
    tweet : "Hackers Accessed Recently Deleted Photos and Files From iPhone X '"
    date : "2018-11-16 00:28:04"
    URL : "https://t.co/qVdt8bet2b"
```

Fig. 8. JSON format of Tweets



```
{ } http://d.opencalais.com/dochash-1/fc8efdeb-48c5-3c13-9cb6-371294e385b8/SocialTag/7
    _typeGroup : "socialTag"
    id : "http://d.opencalais.com/dochash-1/fc8efdeb-48c5-3c13-9cb6-371294e385b8/SocialTag/7"
    socialTag : "http://d.opencalais.com/genericHasher-1/d33d385e-fda7-3112-a7a2-f015a207bb9e"
    forenduserdisplay : "true"
    name : "Apple ID"
    importance : "2"
    originalValue : "Apple ID"
{ } http://d.opencalais.com/dochash-1/fc8efdeb-48c5-3c13-9cb6-371294e385b8/SocialTag/8
    _typeGroup : "socialTag"
    id : "http://d.opencalais.com/dochash-1/fc8efdeb-48c5-3c13-9cb6-371294e385b8/SocialTag/8"
    socialTag : "http://d.opencalais.com/genericHasher-1/ff0c63bb-caa6-3558-ba45-024ef591a5f4"
    forenduserdisplay : "true"
    name : "Social hacking"
    importance : "2"
    originalValue : "Social hacking"
{ } http://d.opencalais.com/dochash-1/fc8efdeb-48c5-3c13-9cb6-371294e385b8/SocialTag/9
    _typeGroup : "socialTag"
    id : "http://d.opencalais.com/dochash-1/fc8efdeb-48c5-3c13-9cb6-371294e385b8/SocialTag/9"
    socialTag : "http://d.opencalais.com/genericHasher-1/4b9dc174-e90c-31b4-9b63-092d4e272935"
    forenduserdisplay : "true"
    name : "Justin Tanner Petersen"
    importance : "2"
    originalValue : "Justin Tanner Petersen"
```

Fig. 9. Open Calais Tags appended in JSON file



```
PS D:\> cd '.\Masters 105061496\Fall 2018\Intro to AI\Cyber Semantic Web\'
PS D:\Masters 105061496\Fall 2018\Intro to AI\Cyber Semantic Web> node .\CSW.js
Hack
-------------------------------------------------
Tweet: Vision Direct hack affects 16,300 customers, exposes payment details  '
Date: 2018-11-20 10:31:03
URL: https://t.co/f0apNytDt3
-------------------------------------------------
Tweet: Microsoft PowerPoint Susceptible to Malware Attacks  '
Date: 2018-11-19 22:20:52
URL: https://t.co/vNT4ADJR6b
-------------------------------------------------
Tweet: Chinese Hack Into Companies for Aircraft Technology  '
Date: 2018-11-04 19:20:04
URL: https://t.co/AoVsOjOz6n
-------------------------------------------------
Tweet: Aaron Smith Sextortion Scam Appears To Leverage On The Necurs Botnet Infrastructure  '
Date: 2018-11-03 12:55:04
URL: https://t.co/J19NefQL5c
-------------------------------------------------
```

Fig. 10. Final Result. Keyword = ’Hack’

Figure 8 presents the output of when the datastore consisting of the cleaned tweets is converted to the JSON format. Figure 9 presents the output of when the JSON file shown in figure 8 is passed through the Open Calais API script. The social tags, importance, score, etc are all appended within the JSON file after being processed by the Open Calais API. Figure 10 shows the result when a specific keyword is entered and the corresponding data is retrieved. The keyword entered is ”Hack” and from our datastore the tweets that are related with that specific keyword are retrieved.

V. Discussion

We perform our initial analysis of our prototype by collecting a plethora of tweets from eminent cybersecurity micro-bloggers. The overall number of tweets extracted are in the range of 6000 to 8000 per blogger. We evaluate the following aspects:

- The overall quality of the tweet after pre-processing.
- The level of the insight that can be gained from the tweet.
- The accuracy of the code to discard irrelevant tweets.

As discussed, before we found that there were many encoding issues related to special characters, retweet labels, emojis, etc. To solve this problem, we analyzed the data and wrote specific regular expressions to remove the impurities within each tweet. After preprocessing, we took 1000 random tweets and cross validated to see if the impurities had been removed. At the end of this step we compared each tweet and found that all impurities had been discarded leading to perfectly developed regular expression which can be used for any tweet extracted from Twitter. After converting the tweets into a JSON format, we passed our tweets into the Open Calais API script to add relevant insight features to our existing Twitter data Store. Initially we saw that the insights generated were of a wide spectrum of topics out of which many were irrelevant with low confidence score. Based on this we set a threshold of 75 percent which is 0.75 in terms of the confidence score value. Any tags having a score below the threshold value were discarded. The last parameter that we evaluate is the relevancy of each tweet. We found numerous tweets which passed the threshold value but were still irrelevant. There were also a few anomalies observed like "@ariannabotaku We strongly apologize for publishing this. We don't publish any kind of ads or promotion kind of art ". This tweet was irrelevant to the data store and still had topics having a confidence score of 79 percent. Finally, we decided to increase the threshold value to 85percent or 0.85.

VI. Conclusion

In this report, we describe our Cyber Semantic Web which gives the end user cybersecurity intelligence for analysis and to serve as an intelligence source. We utilize the Open Calais API to generate contextual tags related to security vulnerabilities with the help of Natural Language Processing. We store the generated social tags, tweets, URL, date, topics, confidence score, importance within a JSON file. The end user can now query the search engine with a search keyword or a phrase. The social tags and tweets will be traversed and all that match the keyword or phrase will be retrieved. The system works under the following assumptions:

- That all tweets that are extracted will be in the English language.
- All the data gathered from the twitter, mainly the tweets from the giant organizations like GitHub, Kaspersky, and The Hacker News report are legitimate sources.
- The contextual information will work better rather than just cosmetic results for a keyword.
- The ontology created is working perfectly and it is the most appropriate for our system.

The main benefit of this system is that it serves as a cybersecurity knowledge base. One drawback of our implementation is that the searching module is not as efficient as it can be. When a phrase is entered certain stop words are not discarded in the search parameter. Overall, in this implementation we have successfully implemented an intelligent search engine which can help you obtain a plethora of cybersecurity intelligence.

VII. Future Work

In the future work, we would like to incorporate an alert module within our system. With this alert module, users can register themselves and specify what kind of alert they would like based on a keyword. If that alert is triggered, users will be notified about the alert and they can view the data that triggered the alert. With this we can also incorporate a user feedback module to improve the quality of cybersecurity alerts that are generated by the system. This can then be converted into a commercial product to be sold to organizations to automate the process of monitoring an organizations privacy and security.

Acknowledgment

We would like to express our sincere gratitude to our course instructor Dr. Ziad Kobti and our research supervisor Dr. Saeed Samet for their continuous support in our project, patience, motivation, and immense knowledge. Their guidance helped us throughout our time of research and implementation of this project. We could not have imagined having better mentors for our study. We acknowledge Parth Shukla for doing tasks: back end implementation, front end implementation, and documentation. We acknowledge Dipesh Patel for doing tasks: back end implementation, front end implementation, and documentation. We acknowledge Manan Patel for doing tasks: linking front end to back end, and documentation.

References

[1] S. Mittal, P. K. Das, V. Mulwad, A. Joshi, and T. Finin, CyberTwitter: Using Twitter to generate alerts for cybersecurity threats and vulnerabilities, in 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), 2016, pp. 860867.

[2] R. P. Lippman et al., Toward Finding Malicious Cyber Discussions in Social Media, in Workshops at the Thirty-First AAAI Conference on Artificial Intelligence, 2017.

[3] Protege Wiki, 27-Nov-2018. [Online]. Available: https://protegewiki.stanford.edu/wiki/MainPage. [Accessed: 27-Nov-2018].

[4] Open Calais, Open Calais, 27-Nov-2018. [Online]. Available: http://www.opencalais.com/. [Accessed: 27-Nov-2018].

[5] Tweepy, 27-Nov-2018. [Online]. Available: http://www.tweepy.org/. [Accessed: 27-Nov-2018].

Appendix

A. Group Work

In our group we utilized an agile software development model, where we decided the amount of work to be performed in an iteration and length of each iteration was around 1-2 weeks. This time length was dependent on each individuals time schedule. After each iteration we used to validate our work based on the task breakdown structure and accordingly plan our next iteration. In this implementation, we did not employ any industrial standard testing methodology because we did not have a vast amount of parameters to validate.