Vulnerability Exploits Advertised on Twitter

Md Tanvir Arafin and Richard Royster

Abstract

The popularity of new social media platforms such as Facebook, Twitter etc. has created an open platform to disseminate information about common software vulnerabilities and vulnerability exploits. Also, there exists speculation about existing markets for vulnerability exploits in these mediums. In this work, we have collected and analyzed about 5,000,000 tweets over 15 days period for understanding the nature of advertisements regarding vulnerability exploits in Twitter. Our study reveals that, vulnerability related tweets are mostly informative and their number has some correlation with the risk presented by the vulnerabilities. We also found code fragments, exploit demo and information dissemination networks related to different vulnerabilities in Twitter. Our limited study period and processing capabilities did not uncover any underground sales of vulnerability exploits; however, we found that, there exist widely followed legitimate corporations that deals with business of selling vulnerability exploits. This limited study should be considered as a stepping stone in exploring vulnerability exploits advertised in Twitter.

Introduction

The world today is leaping forward to real time communications with Facebook, Twitter, and other social networking sites. Twitter has appeared to be one of the most common and preferred social networking medium that allows us to communicate information, follow and be followed by other users. Such communication medium has also made it possible to send information about malicious and non-malicious cyber events in an open forum. So, the questions that we are asking are: (i) Are vulnerability exploits are advertised on Twitter? (ii) If they are then who are tweeting them and who are following? (iii) Can we determine which vulnerabilities are exploited in the wild by analyzing Twitter messages? (iv) Can Twitter analysis be employed to gauge the risk presented by each vulnerability exploit? The goal of this work is to answer these research questions.

To answer these questions, the paper will be broken down to the following sections: section 2 will give background on Twitter, Twitter API, and Common Vulnerability Exposures (CVE) database; section 3 will discuss the process of gathering tweets; section 4 will talk over the processing of the tweets into different classes, section 5 will present the results and findings that we uncovered and finally, section 7 will be our conclusion.

Background

Twitter is one of the many social networking sites that connect people to follow anyone or groups that we wish to connect with. The idea behind twitter is to allow people to blog to a large audience their feelings, thoughts, and emotions. The tweet itself is only 140 characters long. This allows SMS messages to interact with Twitter. So the information relayed is usually small blurts or shorten URLs. Twitter is hitting an all-time high with usage and is only behind Instagram of most popular among certain age groups.

Twitter has Application Programming Interface (API) for developers to interface with twitter. These APIs are used to search through past tweets (REST API)[1] or to gather them while they are happening (Streaming API)[2]. These API do not just return the text of the tweet but also additional information from the metadata associated with each tweet. The REST API uses the resource of GET search/tweets this will return a collection of tweets matching a specified query. The streaming API works along the same concept as REST API where a resource such as POST statuses/filter or GET Statuses/sample. These resources grab tweets that match one or more filter predicates. The Twitter API gives us the ability to sample a small portion of all the tweets being tweeted across Twitter.

To gather data on vulnerability exploits, one good resource is national CVE database. CVE is used to categorize and document vulnerabilities in operating systems and other software that has been released to the public. This list which was started in 1999 has over 58,000 vulnerabilities to date. Most of these vulnerabilities are giving a Common Vulnerability Scoring System (CVSS) number from 0.0 to 10.0. The areas that are evaluated are by three metric groups: Base, Temporal, and Environmental. These groups are broken down even more into characteristics these values are assessed according to risk to each area. The score is calculated using a formula which goes beyond the scope of this paper and is given a rating of High (10.0 – 7.0), Medium (6.9 – 4.0) and Low (3.9 – 0.0)[3]. There are other scoring metrics that track the same areas as the CVSS but we mainly focused on using the scores from CVEs. The name of each CVE is designed as such CVE-YYYY-XXXX[4] the YYYY is for the year and XXXX starts at 0000 and numbers count up to represent each vulnerability when it is disclosed to the public.

For our data collection and analysis, we used the R programming language. R is free and has a GNU public license. The R environment is an integrated suite that performs data manipulation, calculation and graphical display. The language of R is designed around a true computer language that users can add to and define new functions. With this freedom to create a process to gather and identify tweets through the Twitter API we can now have a statistical interaction with the data we uncover.

Data Collection

As discussed above, for data collection we had the option to use either Twitter's REST API or the streaming API. The R-package *twitteR*[6], provides an updated an interface to the Twitter web REST API, that supports Twitter's new authentication system. Another option is to use the *streamR* package [7] to collect data from the streaming API. We use both the package for different analysis.

The first step in our work was to collect tweets that match certain keywords list [The keywords are given in the attachment: base_dictionary.txt]. Although this small list is limited by our initial knowledge on security research, we parsed some valuable results using this small wordlist. The keywords are broken down into the major software, CVE- which is the start of all CVEs,, and code fragments of what could appear in a malicious code. This initial wordlist helped us to stream tweets from the streaming API and process them for further analysis.

From the initially parsed tweets, we uncover the security professional and the communities that are concerned about the vulnerability exploit advertised in Twitter. We advanced our small dictionary of keywords by analyzing the tweets posted by security groups such as Vupen, The Hacker News, FireEye etc. The scripts we use to capture tweets allowed us to run our system for days at a time and collect over 5,000,000 tweets over the span of 15 days. Due to the limitation of our hardware, the tweets we gathered are sporadic in temporal distribution and cover tweets for the following dates: Nov 12-14, 2013, Nov 17-20, 2013, Nov 26-29, 2013 and Dec 4-8, 2013.

Data Processing

One way of handling this large volume of data is to divide into small parts, compute and join the results. We parsed the tweets from the streaming API and saved them in an hourly basis. After accumulating the hourly data, we search the contents of the tweets over an extended dictionary that contains not only the security terms but also some business terms as well [The second dictionary is attached with the report.] Depending on the occurrence of the terms we attached new tag fields with the tweets that record the occurrence of the dictionary terms. We define the cumulative score as the total number of tags a document contains. We found that with the basic vulnerability dictionary, we parse approximately 500000 tweets. However, when we clean the corpus by tagging with an exact search and a better dictionary, less than 4% of the tweets contain nonzero cumulative score. We separate the tweets in new .Rda files. Later, when analyzing different aspects, we search the text, user or tag fields of these files and join the results in a data frame for further processing.

For most of the analysis presented in the work, we used *tm* package [8] of R for text mining. We converted the data frame resulting from our database search to term-document matrices for text mining purposes. We used standard word stemming, stop word removals and case-conversion function to clean our text corpuses.

We understand that, tagging and word based searching has their own limitations and therefore, for better searching and classification of tweets we sought for adapting machine learning approaches. To analyze the effectiveness of machine learning approach in our search, we have used k- nearest neighbor approximation algorithm provided in the *plyr* package of R. k-nearest neighbor is an algorithm used for classification and regression that can predict using a training set to recognize patterns. K-NN is a two phased machine learning algorithm which consists of the training and classification. Training phase is where the algorithm will store the feature and class labels from a training set. The classification phase is where an unlabeled vector is classified through assigning labels based on the training set data.

Due to the time limitation of this study, we trained a small subset of tweets (957 tweets) for analyzing the effectiveness of k-nearest neighbor method. We took random tweets that have a cumulative score greater than 2. We transpose the regarding term-document matrix into document term one and provided a new column that contained the type of the message conveyed by the tweet. We defined three categories of tweets as given in the table below:

Tag	Example
Not relevant/eligible	"#F-SecureGB: Get 30% #Discount Internet Security, Protection For Surfing And Shopping, Using #Voucher Code, Here: http://t.co/wSH10CmKSZ ツ"
Informative	"E.KIA (Enemy Killed in Action) – Microsoft Office Zero Day (CVE-2013-3906) http://t.co/ZcKFsKEi2U"
Malicious claim	After Win7/Srv2008 & BSD, we successfully exploited the Intel x64 Sysret vuln on Xen hypervisor to achieve a full VM to Host escape.

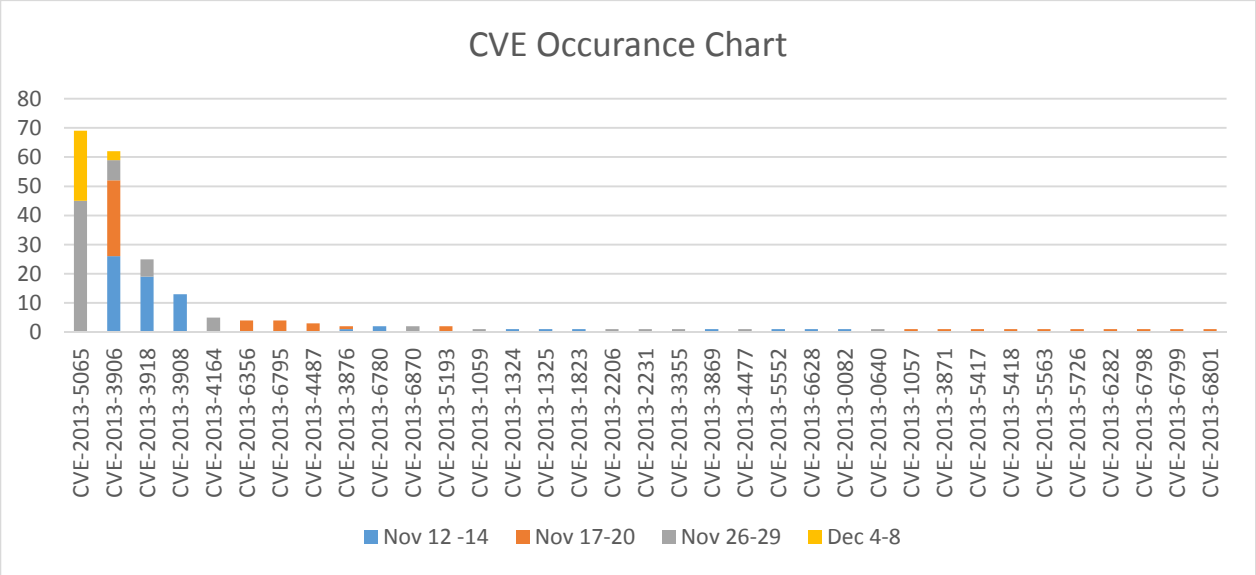
Table 01: Example of different classes of tweets

First, we manually train the tweets about their categories. Then to test the accuracy of this approach, we randomly take 70% of the training data and untag the rest 30% data for using them as test data. We find that in this circumstance the accuracy of the method is about 84%. Several testing shows that, the total accuracy ranges from 70-85%. However, accuracy in detecting a malicious claim is very low (from 0-40%) and accuracy detecting informative claim is moderate (35%-80%). The low rate of finding malicious claim is due to the fact that the number of not eligible class is more dominate than the others. We conclude that knn is weak in differentiating malicious claims from informative tweets; however, a trained knn can become very useful in differentiating irrelevant tweets from the relevant ones.

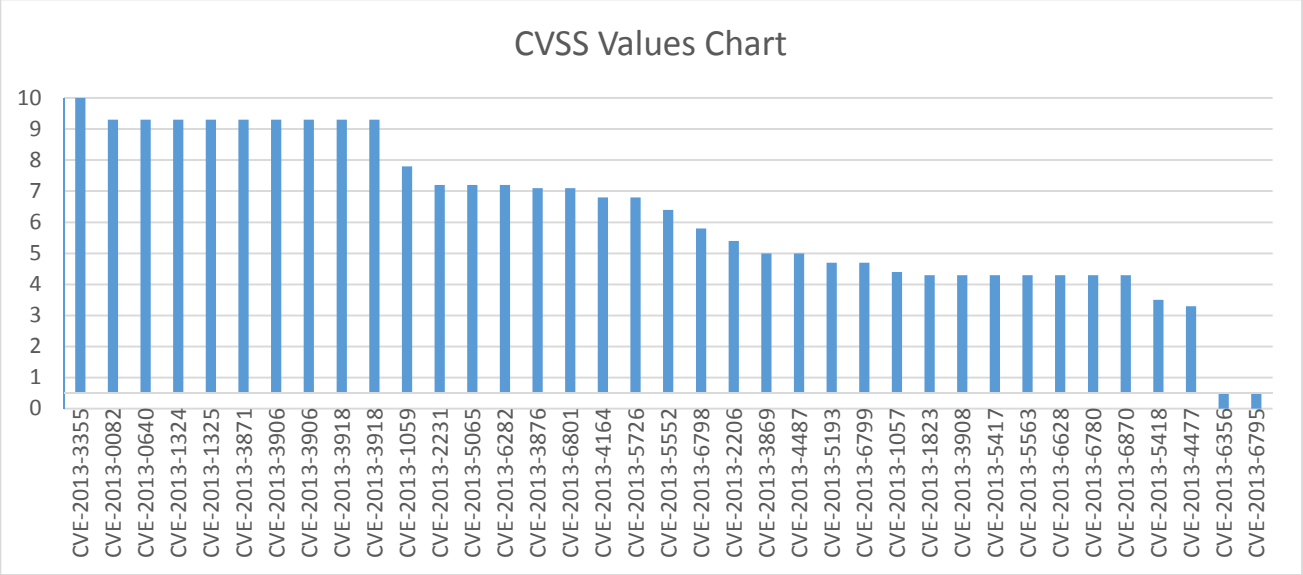
Calculations and Results

Vulnerability appearance on the Twitter

Our first research question was whether vulnerability exploits are advertised in Twitter. We search for the current CVE entries by their exact entry number in the tweet-database that contains tweets having non-zero cumulative score term. The vulnerabilities that make the most appearance in our dataset are shown below.



(a)



(b)

Figure 01: (a) The search result over our complete database for the published vulnerabilities in the CVE database. As the search looks for the exact term "CVE-XXXX-XXXX", we have a very small number of hits per vulnerabilities. (b) The CVSS values chart for each vulnerability analyzed.

It can be seen from figure 01 that, the top 2 pronounced vulnerabilities in our dataset are CVE-2013-5065 and CVE-2013-3906. We will take a closer look at these vulnerabilities and some of the tweets that were associated with them. Also, if we compare both charts, it can be seen that the number of appearances for a given CVE in our database does not have a proportional relationship with its CVSS rank. However, as the analysis looks for only the CVE index, a better search with corresponding terms may provide a better view on the presence of vulnerabilities in twitter space. Also, we can further deduce that, one can have some idea on what vulnerabilities are exploited in the wild by analyzing their signature in twitter-space. We group the tweets regarding these two vulnerabilities by doing a simple search of CVE numbers, related terms and the tags. We the go through the data and manually inspect the resulting data sets.

If we look at the CVE-2013-5065 vulnerability we found that it is a vulnerability in NDProxy.sys in the Kernel of Microsoft Windows XP SP2 and Server 2003 SP2 that allows local users to gain privileges via a crafted application. It was exploited in the wild in November 2013[9]. Microsoft confirmed this exploit on November 27th [10]. Another website that published this vulnerability was fireeye.com, which produces real-time threat prevention to their clients. We see the emergence of this vulnerability in our dataset on November 27th as soon as Microsoft's announcement. After its inception point, we found a huge up-rise in public awareness and CVE database gave it a 10.0score at November 28th. We also found a demo of the exploit in the wild which has been advertised on twitter by twitter name, 0x009AD6, from Japan.

Another important vulnerability exploit was CVE-2013-3609. This was an exploit that allowed remote attackers to execute arbitrary code via a crafted TIFF image. The entry date of this vulnerability was June 3rd, 2013 but does not indicate when a vulnerability was discovered, it was reported being exploited in October. Microsoft first released information about the vulnerability of November 6th. Our search on the REST API and our initial undocumented database contains tweets about the vulnerability right after its announcement from Microsoft. This CVE had a wide public attention and it was still the talked about in early December. Microsoft claims that it had been patched but several twitter accounts (e.g. virusbtn) tweeted otherwise. Also, there were advertises and demos of this exploit available via Yuang Yu. Also, we noticed a change in the language of the tweets regarding this tweets. It started mostly as English, but in the later databases there were a lot of Chinese and Russian tweets that talks about this vulnerability and related topics.

We also found several tweets about the buying and selling of vulnerability exploits by Vupen and ZDI (zero-day initiative). We did not found any underground network of buyers and sellers of vulnerability exploits, however, this might arouse from our limited amount of data collection and analysis. A further study on a broader data set will allow us to locate such market (if there exist any).

In the following table we provide some examples on tweets about vulnerability advertisements:

User	Tweet
TheHackersNews	NSA bought Hacking tools from 'Vupen', a French based zero-day Exploit Seller
0x009AD6	Microsoft GDI+ TIFF Integer Overflow (CVE-2013-3906) 0day Exploitation & Mitigation Demo
deepquest	Novell Zenworks Mobile Device Managment Local File Inclusion Vulnerability http://goo.gl/mjNKD
deepquest	VoipNow <= 2.5 - Local File Inclusion Vulnerability http://goo.gl/tuoEqh

Vulnerability information network

To gain a better idea about the eco-system of the vulnerability advertisements, we search for the retweeting pattern for different CVE vulnerabilities exploited in the past month. From our dataset we deduce several information centers from which common information about the current vulnerabilities are

disseminated. Below, we provide the retweet pattern for CVE-2013-3906, the most tweeted vulnerability for last month.

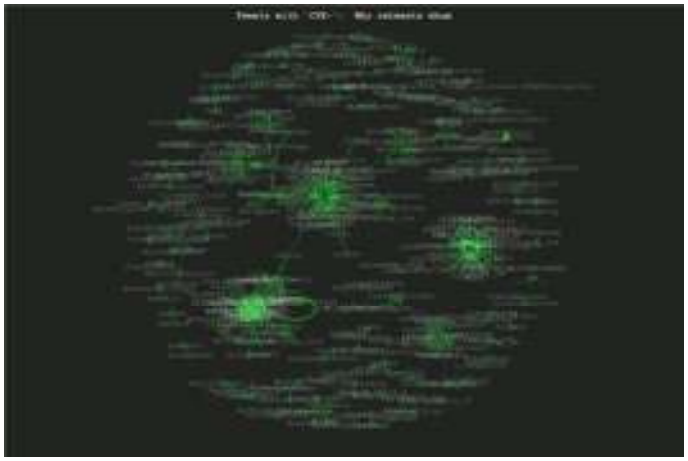


Figure02: Retweet pattern for CVE-2013-3906.

From the distribution network, we have the top 6 account that provided most of the retweeted information. We found that, except for Security Wang, others are established security professionals who usually talks about these exploits as a part of their business.

Account	Followers	% of Retweet population
Fireeye	47801	8.3
The Hacker News	112084	7.1
Security Affairs	4115	5.5
Security Wang	2231	3.6
Threat Intel(Symantec)	19000	3.1
Webroot	12341	2.8

Analysis of tweets from security experts

Contents of vulnerability advertisements are based on the terms associated with current vulnerabilities that are being exploited in the wild. Therefore, for maintaining a better dataset one needs to reseed and refresh the dictionary used for the data acquisition. One way of gaining insight on the vulnerabilities exploited in the wild is to analyze the tweets from the most popular security experts. So, we collected the REST API tweets from four security experts (Vupen, FireEye, The Hacker News and Threat Intel) and tweets advertising CVE vulnerabilities. We group all these tweets and performed text-mining to get the common vocabulary in present vulnerability exploit. The resulting word-cloud is presented below. A quick glimpse on the Vupen tweets informs us about the Microsoft's recent vulnerabilities exploited in the wild. A better study of the top 5% most frequent words gives an idea of vocabularies for most of the vulnerabilities reported and advertised in the given period.

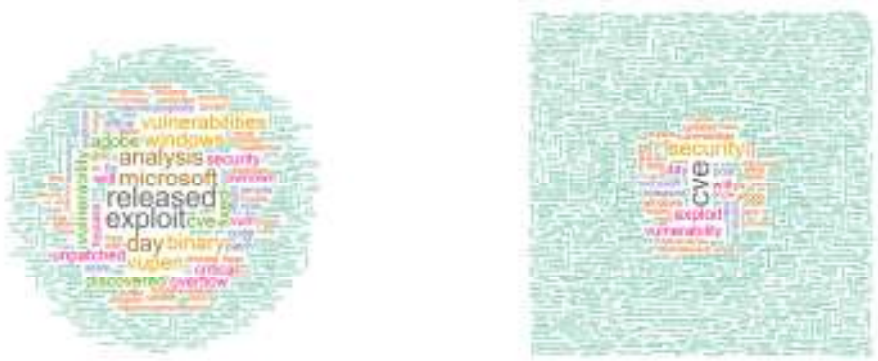


Figure02: (a) Word cloud for *Vupen* for last month, (b) Word Cloud for the most retweeted security experts and "CVE-" contents for last month. The size of a text represents its frequency over the documents. See the uploaded image files for better quality.

Sentiment analysis of vulnerabilities

Finally, we did a sentiment analysis on the REST API data from the first week of November to figure out the effectiveness of such analysis in analyzing public perception on vulnerability exploits on different operating systems. We used the dictionary of positive and negative words by Liu et al. [11] for this analysis. We used the tweets present in the REST API that contains terms related to Microsoft, Apple, Linux and VM system vulnerabilities. We found the general perception about vulnerabilities is usually negative and on Microsoft products this negative perception is highly pronounced. This pronounced negative sentiment might be due to the advent of CVE-2013-3906 which had a comparatively large impact on security concerned or this might be a general consensus about Microsoft's vulnerable products. Therefore, we recommend further studies on these features over a large timeframe to confirm any hypothesis on sentiment analysis regarding vulnerable products.

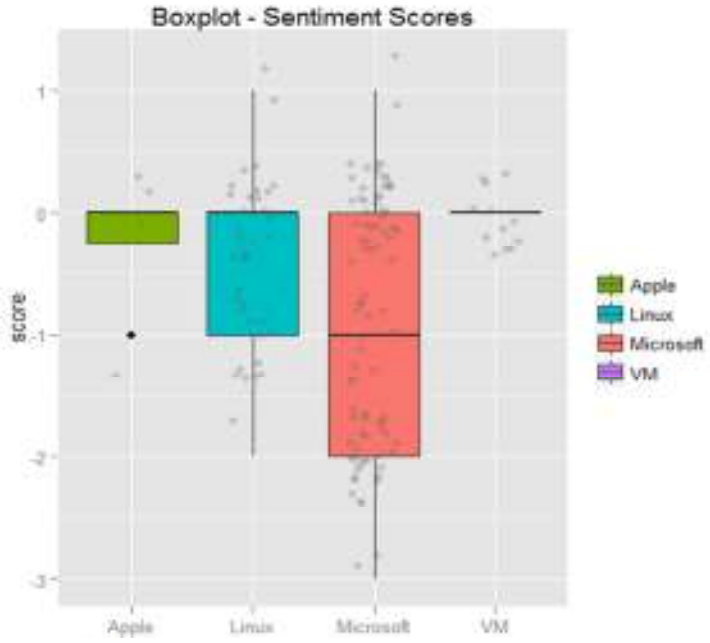


Figure 04: Sentiment analysis over product vulnerabilities of different operating systems.

Conclusions

In this work, we tried to answer some fundamental questions regarding vulnerability advertisement on the Twitter platform. To do so, we have collected about 5 million tweets from the streaming API, developed a dictionary search based scoring technique and K-nearest neighbor based machine learning technique for tweets classification. From our limited search on our twitter database, we found that, there exists some correlation between the CVSS rank and user awareness, and information about vulnerabilities with severe risks is widely distributed in Twitter. We uncover the existence of the information dissemination and retweet network regarding vulnerability exploits and found high centrality in these networks. Our study also provided a better seed dictionary based on the tweets of security professionals. We also found some examples and demos for exploiting vulnerabilities in the wild.

The work suffers from severe limitation in terms data collection since Twitter only allows 1% of random samples on any search topic to be parsed from its Streaming API. We also suffered from the limitation of our hardware and time for large scale implementation of k-NN algorithm. The very simplistic nature of scoring texts is limited to the efficiency and scope of the dictionary provided. From our perspective, this work requires more profound and systematic analysis. The merit of the work is that, it reflects the existence of vulnerability advertisement in Twitter and provides some research direction in terms of dictionary building, user profiling, text mining and improved machine learning techniques that can be employed in similar analysis.

References

[1] https://dev.twitter.com/docs/api/1.1#334

[2] https://dev.twitter.com/docs/streaming-apis

[3] http://nvd.nist.gov/cvss.cfm

[4] http://cve.mitre.org/cve/identifiers/syntaxchange.html

[5] http://translate.google.com/about/intl/en_ALL/

[6] J. Gentry, “twitteR: R based Twitter client” http://cran.rproject.org/web/packages/twitteR/index.html

[7] P. Barbera,"streamR: Access to Twitter Streaming API via R", http://cran.r-project.org/web/packages/streamR

[8] Ingo Feinerer ,Kurt Hornik, “tm: Text Mining Package”, http://cran.r-project.org/web/packages/tm/index.html

[9] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-5065

[10] http://technet.microsoft.com/en-us/security/advisory/2914486

[11] http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

[12] http://krebsonsecurity.com/2011/05/advanced-persistent-tweets-zero-day-in-140-characters/