It's you on photo?: Automatic Detection of Twitter Accounts Infected With the Blackhole Exploit Kit

Joshua S. White
Wallace H. Coulter School of Engineering
Clarkson University
Potsdam, NY 13676
Email: whitejs@clarkson.edu

Jeanna N. Matthews
Department of Computer Science
Clarkson University
Potsdam, NY 13676
Email: jnm@clarkson.edu

***Abstract*—The Blackhole Exploit Kit (BEK) has been called the "Toyota Camry" of exploit kits - cheap, readily available and reliable. According to some estimates, it was used to enable the majority of malware infections in 2012. One major infection vector for BEK is through Twitter. In this paper, we analyze over two months of Twitter data from May through July of 2012 and identify user accounts affected by BEK. Based on reports that BEK infected tweets containing the string "It's you on photo?" were being used to lure victims to BEK infected sites, we identified matching messages and analyzed the associated accounts. We then identified a wider range of message types associated with BEK infection and developed an automated mechanism for identifying infectious accounts - both accounts that were created specifically for malware distribution and legitimate accounts that began distributing malware after the owner's system was infected. Specifically, we find that BEK infectious accounts are characterized by tweets with an entropy lower than 4.5, tweets that are sent using the Mobile Web API and tweets containing an embedded URL. We present an automated method for isolating the point at which an account becomes infectious based on changes in the entropy of tweets from the account.**

I. Introduction and Background

The Blackhole Exploit Kit (BEK) is a web-based application that manages the installation, command, and control of malware. It works by utilizing a compromised server to which victims are directed by means of various infection vectors [1]. Links luring victims to a compromised server are typically distributed through spam email messages, spear-phishing or links in social network posts. The compromised exploit server hosts an innocuous looking webpage that contains a tool for scanning the visiting system's vulnerabilities [7]. When the victim clicks on a link spread through any infection vector, this tool runs on the victim's system.

Once the tool identifies vulnerabilities, BEK automatically loads the necessary exploit tools and then compromises the victim's system. BEK can be used to install a wide variety of malware tool-sets [2] [3] depending on the exact mission of the attacker. BEK contains modular capabilities that allow new exploits to be added rapidly and in many languages. BEK employs countermeasures such as packing of its malware payloads, binary obfuscation and antivirus avoidance capabilities.

According to some estimates, BEK was used to enable the majority of malware infections in 2012 [39]. One study found that BEK accounted for 29% of all malicious URLS in a dataset consisting of 77,000 URLs marked harmful by the Google Safe Browsing API [7].

In early 2012, the creators of BEK released version 2.0 and with this release, it became one of the most well known and most commonly deployed exploit kits in existence [4] [5] [8]. Version 2.0 was a complete rewrite of the toolkit and included various improvements, such as new techniques for antivirus detection avoidance, multiple operating system support, and better command and control capabilities.

With past versions of BEK, victims were required to visit a webpage with the naming convention *protocol infection_server.TLD(letter).php*, where "letter" was any English lowercase alphabetical character. This naming convention was suspicious to many potential victims and as such was dropped in version 2.0 [6]. BEK 2.0 allows attackers to specify custom created URLs based on either a hard-coded schema or one which includes variables. These dynamically created URL's can be made so that they are only valid for a short period of time or even limited to the duration of a single client session. With these changes, defenses based on URL blacklisting techniques become ineffective against BEK.

BEK continues to be prevalent and feature rich due to both its modular design and the consistent level of maintenance given by it creators. The level of effort provided by the creators is fueled by the Malware as a Service (MaaS) economic model [12]. BEK is optimized to deliver its intended results for a paying attacker. The Blackhole Exploit Kit 2.0 regularly sells annual licenses for $1500 US dollars, with the option of renting the kit for $50 a day [7]. It is believed by many that BEK was created by a Russian criminal group [9]. Though some speculate that it may be only heavily used by this group, but not specifically created by them [10].

BEK includes countermeasures against many common forms of defense, including:

(1) Email Spam Filters Email spam filters rely on known spam message structure, key words, and URLs to detect suspicious messages [13]. BEK overcomes this by duplicating legitimate email messages which were captured from real compromised accounts. In addition, URLs point to one of many thousands of compromised web servers including those running on legitimate hosting services. At one point, a single day's campaign launched by an attacker using BEK, utilized

1960 URLs pointing to 291 infected web servers [1] [11].

(2) User Education Educating users to identify BEK messages requires that the user be able to detect suspicious messages via visual inspection of link addresses or message content/grammar. This is typically effective only when a message is automatically generated. In the case of BEK, messages were cloned from legitimate ones.

(3) Reputation systems Ranking systems deployed by Twitter, Google and others rely on links being active long enough to be classified. BEK URLs have a limited time to live and may not get scanned before being automatically disabled. In addition, BEK landing pages employ various JavaScript obfuscation techniques to better hide their capabilities. Documentation that accompanies BEK lists the following feature:

> "Implemented maximum protection from Automatic systems for downloading exploits, used by AV companies: generate a dynamic URL, which is valid for a few seconds, you need only to [lure] one victim at a time [11]."

In this work our focus is on the BEK infection vector itself rather than on the final installed malware payload. Specifically, we analyze how BEK uses Twitter as an infection vector as it is the most common infection vector used by BEK[6].

The rest of the paper is organized as follows: Section II describes our dataset and how it was collected. Section III discusses our analysis methods and several key metrics. Section IV presents the evolution of and the results of the our automated mechanism for the detection of infected accounts. Section V compares this work to prior published works in the area of Twitter as an infection vector and Section VI concludes.

II. Dataset Characteristics and Data Collection

Over the course of 2012, we collected multiple terabytes of data from Twitter. Recently released estimates place total Twitter traffic at 175 million tweets per day [15]. Comparing the amount of data we collected daily to Twitter's own reports on traffic per day [15], we estimate that we collected between 50% and 80% of all Twitter traffic.

Our complete 2012 dataset consists of 147 Days of Twitter data stored as 30 Terabytes of gzip compressed JSON formatted data. A typical day's compressed data can vary significantly, but averages 70 GB.

For this analysis, we focused on the primary infection time frame of May 3rd through July 7th of 2012, although we collected substantially more data throughout 2012 and 2011. This time frame includes the date where the first infectious account was detected, through the point where more than 1000 accounts were affected. Our 2 month sample set consists of 6,531,319,202 Tweets, with 265,163,290 unique accounts represented.

When processing a stream of Twitter traffic, each tweet contains both the 140 character message and a rich set of metadata describing the origin of the tweet. We captured complete tweet data in JSON format using Twitter's REST API. This data includes a large number of additional fields other than the message text, all of which can be taken into account during analysis. This additional metadata, includes information about the user, the users account, and potentially the users location at the time of the tweet. Since Twitter is a social networking site, connections between users can be gleamed from the included post data, for example if the message is being retweeted or sent directly to another user. Together, the message and its associated metadata for each tweet averages 1.2 Kilobytes.

The tweet message and associated metadata is returned as a JSON string containing key-value pairs. The fields or keys of this JSON string are listed in Table I. Any of these items could potentially be used as indicators of an infected account if measured and compared properly. Aspart of our analysis, we will discuss which items have the strongest correlation with BEK infection.

TABLE I. Twitter JSON Key Fields

profile_link_color	Coordinates
In_reply_to_screen_name	Geo
In_reply_to_status_id	text
In_reply_to_status_id_str	entities
In_reply_to_user_id	place
profile_background_color	contributors_enabled
profile_background_title	default_profile
default_profile_image	description
follow_request_sent	followers_count
friends_count	geo_endabled
profile_image_url_https	listed_count
profile_background_image_url	notifications
background_image_url_https	name
profile_image_url	lang
sidebar_border_color	use_background_image
sidebar_fill_color	screen_name
profile_text_color	show_all_inline_media
url	utc_offset
Created_at	Id
Favorited	Id_str
retweet_count	created_at
favorites_count	following
id_translator	location
trunkated	retweeted
Contributors	protected
time_zone	statuses_count
verified	

When collecting this data set, we encountered and overcame a number of interesting challenges. The foremost challenge was dealing with changes in the way that Twitter limits the rate at which data can be gathered. We first began gathering Twitter data in January 2011 at this time there were no specific limits on the rate of data available. Twitter first introduced limits to the API live stream in mid 2011 [22] in preparation for licensing their feeds to GNIP, who could then re-sell them at approximately $30,000 a month [21]. This change meant that normal users/researchers could no longer access the Firehose (ie. 10%) stream directly, and no one could be whitelisted for the 30-40% stream.

After this change, researchers and other non-commercial users still had access to the Spritzer service and the limited streaming Twitter API. The Spritzer feed is approximately 1%

of all tweets happening in real time. The Streaming API limits this further by allowing 350 requests to Twitter an hour, with a response of no more then 200 tweets [23] [24] [25]. That's a total of 70,000 tweets an hour per application. An application is denoted as any tool written and distributed to multiple users, but using the same account credentials. Additionally, Twitter imposes a cap of 20,000 tweet responses an hour for any single IP address. Together, this limitation means that based on a current reported rate of 175,000,000 tweets occurring daily, the average application, user, IP combination receives no more then 480,000 tweets daily (20,000 * 24 hours), or .27% of all tweets.

Twitter uses a fairly simplistic method for sampling the live stream [41] for the Spritzer Streaming API. Specifically, it starts a counter for each new connection. This counter starts at 1 and counts to 100 before starting over. The first tweet "1" is sampled and returned to the connecting client. After some experimentation, we discovered that putting in a simple "sleep 1" command between the start-up of two or more subscriber scripts is enough to receive a completely different set of tweets between the accounts. To prove this point, we initially created 5 Twitter Accounts and manually started the collection script with each set of login credentials one after the other. After running for 24 hours, these 5 Subscriber Script / Account combinations collected 16,599,674 Unique Tweets. Following this logic, we utilized 30 unique accounts for the collection period to generate our sample dataset.

III. Data Analysis

A. Analysis Framework

Our data analysis system consists of a small cluster consisting of 18 x 64 Bit Cores with 52 GB of RAM and 48 Terabytes of storage capacity. We distribute our dataset over the cluster using DISCO Distributed File System (DDFS) [18]. DDFS is a non-traditional tag-based filesystem designed for use with the DISCO Map Reduce system. DDFS builds on top of a traditional Linux filesystems like ext4, adding horizontally distributed scaling capabilities.

We store the compressed data in three external USB RAID enclosures consisting of approximately 11 TB of storage each. For analysis, we move portions of this data to an 18 TB RAID array for processing.

We use the DISCO [17] framework for creating, maintaining, and distributing jobs written in Python. The Twitter JSON stream enters the system and is broken into chunks for each day and then within each day, it is further broken into slices that are limited to 64 MB in size. DDFS refers to chunks with the name (blackhole:Date) and slices with the name (Date$SliceNumber) where chunk number is the MD-5 checksum of the data in the slice.

Key-value pairs are derived from the Twitter APIs JSON structure as previously listed in Table I. Data feeding the reduce phase is noted as (k{v, d, n..}) where k is the key that we have worked on and v,d, and n are values, derived values and number of occurrences, respectively. For each query over the data set, we specify the makeup of the key based on the fields available in the JSON structure. For example, if we wanted a query to analyze the application that generated the tweets, we might base the key on the *source* field of the JSON data. An iPad user might generate tweets with a "source" key that looked like: *¡a href="http:twitter.com#downloadipad" rel="nofollow"¿Twitter for iPad¡a¿*. This value tell us three important things: the first is that the tweet was posted using the official Twitter application, the second is that the application was installed on an iPad, and the third is that when that tweet was sent nofollow was enabled which means that search engines will not use it to gauge popularity of a site. These two items are what we refer to as *derived* values. Finally, we sort data from all nodes on the cluster back on the master node by passing the key-value group (k-vg) option to DISCO and feeding it as a sorted list (k-vg)sorted(iter)), in which each key has the cumulative sum of its values.

Disco treats each MapReduce job as a single entity and does not run multiple jobs simultaneously. Since we have a live data stream to process which requires multiple things to happen simultaneously, we do the following: 1) We have a MapReduce job that sorts the data by accounts. 2) The sorted accounts are then written to the DDFS store based on account name. This is similar to creating a new MySQL Table for each username. 3) The data in the DDFS store is then processed for characteristics like the entropy of the message, the API used, etc. 4) A new DDFS entry (like a new row in MySQL) which refers to the users account is created. 5) A count of the number of BEK infectious messages for each account is updated in a Python list.

B. Key Metrics

Two specific values we compute over our dataset using this MapReduce framework are an Entropy value and Pearson's Correlation Coefficient. We will explain in more detail later in the paper the significance of these metrics. In this section, we simply describe how they are implemented in Python for use with DDFS.

1) Entropy: We use entropy as a numerical signature of a tweet's message text. Entropy gives us the character distribution of a message. It is a measure of uncertainty in the random variable, character values [40]. Entropy is a fundamental information theory metric used in statistical Natural Language Processing (NLP). Classical NLP requires the use of machine learning classifiers which would be computationally expensive for the size of the dataset we are using. We instead use Shannon's Entropy formula which calculates the minimum number of bits needed to represent a piece of information, in our case, the message data in each tweet. Without this, we would have had to rely on regular expressions or a natural language tokenization system to look for similarities across text.

Shannon's Entropy formula is given in Equation 1 where n is the total number of characters in the a tweet message and x_i is the ith character. For each character, we take the log of its base2 representation and multiple by its probability within the tweet.

$$H(X) = -\sum_{i=1}^{n} p(x_i) log_b p(x_i) \quad (1)$$

$H(X)$ represents the minimum number of bits needed to encode a string. If we take Shannon's Entropy of the string "It's you on photo?" we end up with an H(X) equal to 3.47135. Since this is not a real number we round up to 4 bits. This means that we require 4 bits to encode each ASCII symbol in our string.

2) Pearsons Correlation Coefficient: Pearsons Correlation Coefficient(PCC) or Pearsons Product-Moment Correlation Coefficient allows the calculation of linear association strength between two variables. In our case, the variables are keys from our Twitter API JSON data such as *text* or *url*. PCC is simply an attempt at drawing a line that best fits through the data a graph of a two variable dataset. This is known as the r-value which is shown in Equation 2.

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y} \qquad (2)$$

$\rho_{X,Y}$ is the Correlation Coefficient between X and Y. This tells us that as the score of X increases or decreases, Y does so at the same rate, either positively correlated or negatively correlated. Values close to 0 indicate that the correlation - either positive or negative - is a weak one.

IV. Results

When analyzing our data set, we compare a baseline of "clean" Twitter accounts to a set of infectious accounts. To identify accounts affected with BEK, we started with reports that BEK related messages had been found in Twitter with the string *It's you on photo — It's all about you?* [19] [20]. We manually examined a subset of the accounts that generated messages matching this string and identified additional message attributes common to these accounts. Based on what we learned through manual inspection, we designed a filter to automatically classify accounts as infectious and apply this filter to the complete two-month data set. We demonstrate the effectiveness of this filter by comparing the set of accounts classified as clean to those classified as infectious. The following subsections explain each of these steps in details and give an overview of our findings.

A. Identification of "Clean" Accounts for Comparison

We identified as set of "clean" Twitter accounts to which we could compare our measurements of infectious accounts. Accounts were selected using Python's random.shuffle() and random.sample() methods. Python's random.shuffle() mixes the population, while random.sample() returns a specified number of items from a population stored in a list. We then manually inspected the accounts to verify that they represented non-advertising, non-infectious, normal user accounts for English speaking users.

We sampled 100 such accounts from our dataset, manually verifying each for content. Our total clean sample set consisted of 197,237 messages, all of which were stripped of any links, hashtags "#" or directed entity names "@". We removed hashtags and entitynames because the variety of names used by users of Twitter is so great that they skew the entropy measurements. Some entities will use names as short as 3 initials, while others use an entire phrase. Measurements have additionally shown that the sheer volume of URL redirection and shortening services in existence, including Twitter's own .to service, prohibits the use of any sort of URL based measurement.

The messages for each "clean" account were sampled using the GetUserTimeline() API option to pull up to 2000 Messages per account over the time period of May 1st to December 31st, 2012.

B. Initial Identification of Infectious Accounts

Our first pass at identifying infectious accounts was to pass over all sample messages with a regular expression that looked for the common message format of (@followername It's you on photo? url.ru/#followername.html) [19]. This format was widely reported starting in July of 2012 and included a variant message "It's about you?." The basic format was used to generate the regular expression shown in Code Block 1.

Code Block 1. BEK Message Detection Regular Expression

```
if re.match( r'(.*)@(.*).ru/(.*)#(.*)html(.*)', text, re.M↩
    |re.I):
```

This returned over 1000 Matches from our overall dataset. After manually inspecting 10 suspect accounts we found a number of interesting, undocumented, BEK infectious message attributes. First and foremost, the assertion that many security related blogs made about message structure [19] [20] was incomplete. Table II is a list of some additional message variations we observed.

TABLE II. BEK Infectious Message Variations

You were nude at party) cool photo)	It's photo of you?
Amazing! your nude photo	It's all about you?
Wow! your photo is cool.	It's about you?
At party you was drunken) cool photo)	It's you on photo?
Your photo is amazing	WOW! you look good)

In addition to the message variations shown in Table II we found a number of other interesting characteristics. First, BEK messages contained various abbreviations of words such as 'ur' for "your". Second, many messages did not include the directed at "@" follower name. Third, the majority of links used a URL shortening service, did not include the follower name, or did not end with .html.

C. Comparing the Entropy of Clean and Infectious Accounts

In Figure 1, we show a graph of the entropy measurements for messages sent from the "clean" Twitter accounts we identified earlier. This graph suggests that normal messages from English speaking users on Twitter have a message entropy between 4.5 and 7.5.

Figure 2 graphs the entropy for an account that posted both infectious and clean messages intermixed over time.

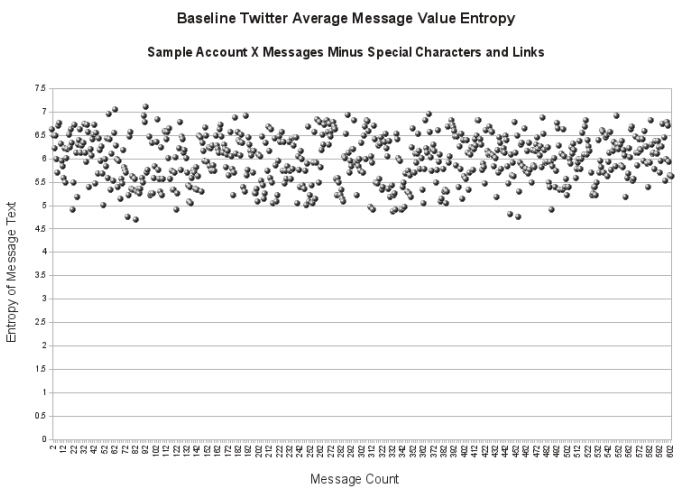

Fig. 1. Plot of Baseline Clean Twitter Account Entropy

We set a group of messages with entropy in the normal range of 4.5 to 7.5, but we also see a substantial group of messages with entropy below 4.5. The outlying values with an entropy near 0 are simply messages consisting of a directed at username and/or URL with no actual textual content, but all other datapoints with an entropy lower than 4.3 match the characterization of BEK messages discussed above.

We examined a set of accounts with this mixed behavior and find two patterns. In some cases, it appears that the accounts have been hijacked, as such it consists of older clean messages and newer infectious messages. In other cases, it appears that the account was created specifically to be an infection medium and includes both infectious messages, and retweets of legitimate messages and/or copied messages from other accounts. This simple method of account obfuscation, mixing infectious messages with retweeted messages, hides the true nature of the account from many methods of detection.

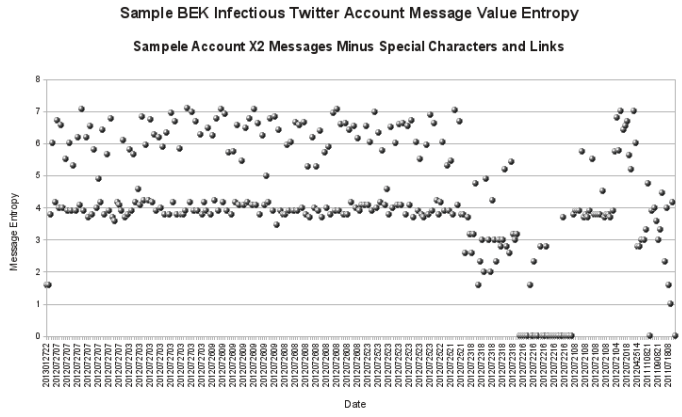

Fig. 2. Plot of Baseline Infectious Twitter Account Entropy Versus Time

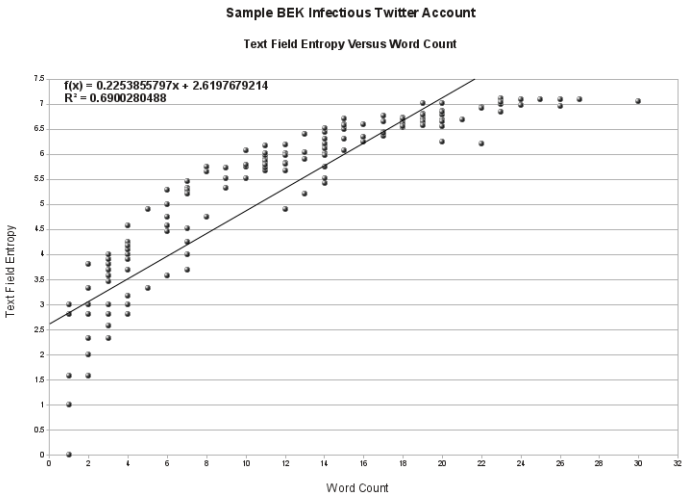

Fig. 3. Plot of Text Field Entropy Correlation With Word Count for Sample Infectious Twitter Account

D. Comparing the Pearsons Correlation Coefficient of Clean and Infectious Accounts

We calculated the Pearsons Correlation Coefficient (PCC) of both our ”clean” Twitter accounts and the accounts we have identified as BEK infectious accounts. When comparing two infectious sample accounts to each other, we have an average PCC value of 0.927581013955. This positive value near +1 indicated a strong positive correlation or similarity between the accounts. When comparing the infectious accounts to clean accounts, we have an average PCC value of -0.0847935420003. This negative value near 0 indicates that the accounts are dissimilar. Figure 3 shows a correlation plot of the accounts we have identified as infectious.

E. Use of the Mobile API

We also searched the contents of the JSON tweet structure for metadata fields that were correlated with BEK infection. We find that 76.7% of all traffic from BEK infectious accounts uses the Mobile Web API to send messages. In addition we find that non-BEK infectious accounts had a 68.6% usage of the standard Twitter Web API.

Figure 4 shows a comparison of source application usage between BEK infectious and normal Twitter user accounts. For this test, we took a sample of 100 BEK infectious user accounts and 100 non-infectious user accounts. We then counted the number of source applications used for each account. The source application field in the Twitter API returns a value which specifies the name or type of application that was used to generate the tweet. Our analysis found a number of different applications which we separate into four (4) groups: 1) Mobile Web, 2) Web, 3) iPhone, 4) Other. Note the substantially higher use of the Mobile API by the BEK infectious accounts.

We suspect that BEK uses the Mobile Web API because it is substantially simpler to use than the normal Web Twitter API. APIs like the Web or iPhone versions require Oauth authentication. This requires registration of the application

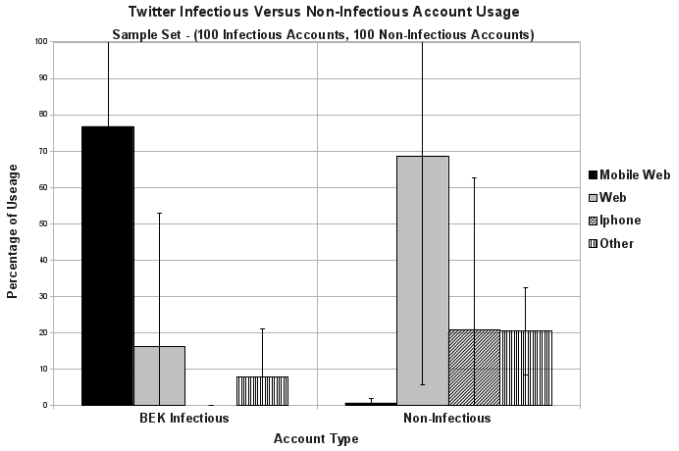

Fig. 4. Source Application Percentage, Infectious Versus Non-Infectious Accounts

with Twitter to obtain cryptographic keys. In addition, the Mobile Web API requires substantially less code to implement.

We searched for other indicators of BEK infection in the JSON metadata, but found no other correlation of message or metadata values. For example, geographic indicators such as latitude and longitude, location tags, and time zones do have a correlation with BEK infection.

F. Summary of Infectious Indicators

We make the following statement about BEK infectious accounts. BEK accounts can be effectively detected by looking for accounts where:

- a percentage of there tweets have an entropy lower than 4.5
- messages that have entropy's lower than 4.5 were sent from the Mobile Web API
- tweets have a URL embedded in them

We use a filter based on these indicators both to identify infectious accounts and to identify the point at which an account becomes infectious. Identifying the point of infection requires access to samples of the account timeline before it became infected so that we can observe the change. Real-time identification of accounts as they become infectious could be quite beneficial for stopping the spread of infections in Twitter.

During our sample analysis of infectious accounts, we came across two accounts which had always been infectious from the time they were created until the time when we sampled them. In other words, our data set includes the creation of these accounts and messages sent from that time forward. These accounts displayed the same characteristics as an account that had only become infectious over time, such as use of the Mobile Web API and message entropy values lower than 4.5.

G. Totals

We processed our entire dataset again with a filter based on the infectious indicators described previously. and Table III summarizes our findings.

TABLE III. SUMMARY OF RESULTS

Total Tweets Processed	6,531,319,202
Total Number of Unique Accounts	265,163,290
Number of Suspicious Accounts	729,609
Total Number of Suspicious Tweets	8,286,480
Calculated Percentage of Infectious Accounts	0.275%
Calculated Percentage of Infectious Tweets	12.7%

The overall percentage of BEK infectious accounts is relatively low overall compared to our dataset. However, 729,609 infectious accounts is still a strikingly large absolute number. BEK uses individually infectious twitter accounts to transmit its messages, but does so by infecting a physical computer and stealing the users Twitter credentials. This suggests that there are potentially 729,609 BEK infected computer systems. We believe this to be a conservative estimate of BEK infection.

We would like to be able to report a definitive false positive rate for our detection mechanism. Calculating an absolute false positive detection rate would require contacting each Twitter account owner to verify whether they generated these messages by hand and perhaps by running a detection program on their computer to see if it is infected with BEK. In many cases, this would be impossible due to the short lived nature of the accounts. However, we are interested in trying to do this manually for a few cases in the future.

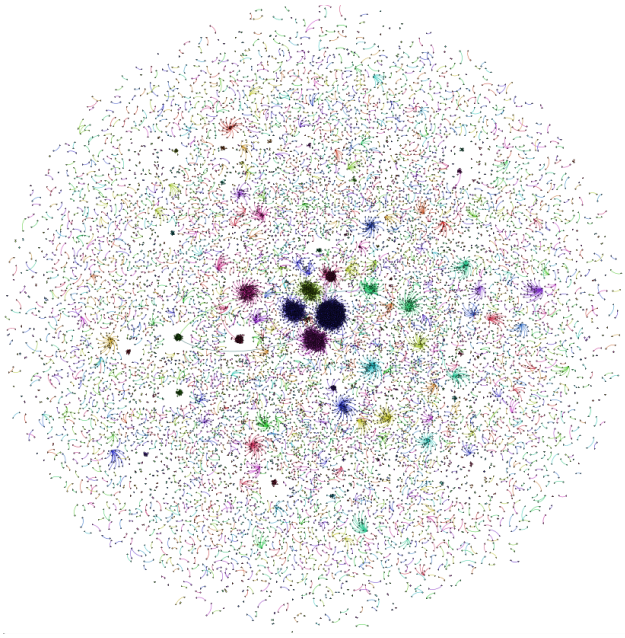

Fig. 5. Clustered Analysis of Infected Accounts

We further visualize the 729,609 suspicious accounts in Gephi [42] using the OpenOrd plugin [43] which show us

obvious clusters as shown in Figure 5. The linkage between the core node of a cluster and others is representative of an infection path where one user infects another. We classify the second account as infected when it has sent messages that conform to our filter. This graph does not include lines representing infectious messages that an account sends when there is no evidence of a resultant infection. The most densely populated portions of the graph represent accounts that have successfully spread their infection to others and as such can be considered infection hubs.

V. Related Work

There is no shortage of work relating to analysis of sites, such as Twitter, to gauge the mood of a population and use these measurements to predict external events, such as stock market trends [26] [27] [28]. In addition, social networking trends have been used as indicators of real world contagion outbreaks, such as that discussed by Manuel Cebrian [29]. Cebrian built on the work of applications, such as Google's Flu prediction system, to include an understanding of the relationships that occur in these networks of outbreaks. In this work, he was able to conclude that location within the network was crucial to the chance of infection. While we did not pursue measurements of group centrality in this work, as relates to BEK infectious account spread, this is an area we intend to pursue in the future.

A few studies on the mechanism that BEK uses for dropping malware onto a system have recently emerged, which add to the various studies on driveby downloads. Grier et al. [7] examined what they term the "the emergence of the exploit-as-a-service." This included a comprehensive analysis of 77,000 URL's and over 10,000 unique binaries. Their findings showed that BEK accounts, in conjunction with the Phoenix Exploit Kit, accounted for around 47% of all pages serving exploits. We found the URL patterns identified in their work, ie: "w.php?f=(.*?)&e=(.*?)" to not be all inclusive, based on our analysis.

In addition, the work of Rajab et al. [30] presented the results of 240 million pages over a year long period, in which they studied malware which masquerades as fake antivirus software. This is a popular method that BEK employs post link visitation from Twitter [33] [34]. Another study, Li et al. [8] presented a study of malicious web advertising, penned malvertising. This work found more than 1% of 90,000 well maintained pages had been exploited and were serving up driveby downloads, malicious links, and various click-jacking based threats.

We have seen various approaches involving the direct analysis of social media networks, specifically Twitter, for malicious content, such as spam and bots. Chu et al. [31] presented an approach to determining if an account was being run by a Human, Bot, or Cyborg based on various factors such as an entropy component which looks for periodic or regular timing of posts. Our work takes a similar approach to theirs in that we look at various account properties. In contrast to their work, we chose not to employ Google's Safe Browsing API [32] to check URL's because this component does not work for live analysis; the GSB is only updated when Google detects or gets a report of a malicious URL.

Along the same lines as Chu et al., other research has focused on the identification of automated users such as that by Zhang et al. [37]. They presented a method for the detection of automated behavior in a Twitter account. They showed that not only do 16% of Twitter accounts show signs of automation, but also only a relatively low number of tweets use the Mobile Web API for posting.

Moore et al. [36] presented the results of a investigation of search term trending abuse. They used top search engine search term results and compared them to Twitter top results. Their study dataset was gathered over a 9 month period and resulted in heuristics for the identification of advertisement sites. They provided statistics on the prevalence of these sites, and measured how seach engines, such as Google, detection and blocking attempts have affected trending term abuse. The authors presented a very informative analysis of CDF (Cumulative Distribution Function) versus MFA (Made For AdSense) domains. These MFA's directly correlate to top Twitter search term results. The same type of external correlation may be possible between social networks.

Unlike these papers, we focus on the potential to build an automated system which does not rely on any external API's besides the Twitter data feed itself in order determine if a specific Twitter account is infectious. We believe that this work can be applied to various other social networking sites with some modification. In the future, we would like to include results from Han et al. [35] on the normalization of social media text. If used, this could potentially solve some problems involving message variation.

VI. Conclusion

We have completed a large-scale analysis of the characteristics of Twitter accounts being used as Blackhole Exploit Kit infection vectors. In some cases, these accounts appear to have been setup for the sole purpose of malware distribution, but more often are simply accounts which have been taken over after infection.

We found that there is substantial variation in message structure when attempting to use simple regular expressions to identify infectious messages. We identify a larger range of message types associated with BEK than we see discussed in other published analysis of BEK [19] [20].

We have identified the aspects of individual tweets that correlate most strongly with BEK. We have used these measurements to construct a filter based on message entropy, the Tweet source (Mobile Web API) and the presence of an embedded URL. Using this method, we identify over 700,000 infectious accounts which have generated upwards of 8 million infectious messages.

We have discussed the integration of measurement techniques into a our large scale social network analytic platform as part of a series of MapReduce functions.

There are likely many more threats within our dataset, and assuredly more on Twitter as a whole. We will continue to develop our platform adding filters and capabilities until it is capable of real time analysis of a wide variety of events and threats.

REFERENCES

[1] J. Oliver, S. Cheng, L. Manly, J. Zhu, R. Paz, S. Sioting, J. Leopando. "Blackhole Exploit Kit: A Spam Campaign, Not a Series of Individual Spam Runs, An In-Depth Analysis," Trend Micro Incorporated Research Paper, 2012

[2] Jason Jones. "The State of Web Exploit Kits," HP DVLabs, 2012

[3] Gabor Szappanos. "Inside The Blackhole," SophosLabs, 2012

[4] Fraser Howard. "Exploring the Blackhole Exploit Kit," Sophos Technical Paper, March 2012

[5] Ziv Mador. "Exploiting Kits: The Underground's Weapon of Choice," Infosecurity Europe 2012, SpiderLabs at Trustwave, 2012

[6] Stephanie Reetz. "Blackhole Exploit Kit 2.0," Threat Analysis from the Multi-State Information Sharing & Analysis Center, September 26, 2012

[7] Chris Grier, Lucas Ballard, Juan Caballero, et al. 2012. Manufacturing compromise: the emergence of exploit-as-a-service. In Proceedings of the 2012 ACM conference on Computer and communications security (CCS '12). ACM, New York, NY, USA, 821-832.

[8] Zhou Li, Kehuan Zhang, Yinglian Xie, Fang Yu, and XiaoFeng Wang. 2012. Knowing your enemy: understanding and detecting malicious web advertising. In Proceedings of the 2012 ACM conference on Computer and communications security (CCS '12). ACM, New York, NY, USA, 674-686.

[9] Artem Gololobov. "Blackhole Expoit Kit," Websense Security Labs Blog, February 2011

[10] Brian Krebs. "Crimeware Author Funds Exploit Buying Spree," Krebs On Security Blog, Jan 2013

[11] J. Oliver. "Blackhole Exploit Kit Spam Runs in 2012," Proceedings of RuxCon 12, October 2012

[12] Pierluigi Paganini. "The Rise of Malware as a Service (MaaS)," HPlus Magazine, February 2013

[13] Gordon V. Cormack, Jos Mara Gmez Hidalgo, and Enrique Puertas Snz. 2007. Spam filtering for short messages. In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management (CIKM '07). ACM, New York, NY, USA, 313-320.

[14] J. White, J. Matthews, J. Stacy. "Coalmine: an experience in building a system for social media analytics," Cyber Sensing 2012, Proceedings of SPIE 8408/84080A(2012) , April 2012.

[15] Shea Bennett. "Just How Big Is twitter In 2012 [INFOGRAPHIC]," All Twitter - The Unofficial Twitter Resource, February 2013

[16] Jeffrey Dean and Sanjay Ghemawat. 2004. MapReduce: simplified data processing on large clusters. In Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation - Volume 6 (OSDI'04), Vol. 6. USENIX Association, Berkeley, CA, USA, 10-10.

[17] Prashanth Mundkur, Ville Tuulos, and Jared Flatow. 2011. Disco: a computing platform for large-scale data analytics. In Proceedings of the 10th ACM SIGPLAN workshop on Erlang (Erlang 11). ACM, New York, NY, USA, 84-89.

[18] Nokia Corporation. Disco Distributed Filesystem - Disco v0.4.4 documentation, Dec 05, 2012.

[19] Graham Cluley. "Outbreak: Blackhole malware attack spreading on Twitter using "It's you on photo? diguise," Sophos Naked Security Blog, July 27, 1012

[20] Rob Waugh. "It's you! Blackhole vierus spreading rapidly via Twitter fools users with fake photo link," MailOnline Science and Tech New, July 2012

[21] @Rsaver, Twitter + Gnip Partnership, Twitter API Announcements on Google Groups, November 17, 2010, URL = https://groups.google.com/forum/?fromgroups=#!topic/twitter-api-announce/4KIAawaY-IA

[22] Christina Warren, Measureing Social Media: Who Has Access to the Firehose?, Machable.com, March 13 2011, URL = http://mashable.com/2011/03/13/sxsw-smaroi/

[23] Mike Melanson, Twitter Kills the API Whitelist: What it Means for Developers and Innovation, February 11 2011, URL = http://www.readwriteweb.com/archives/

[24] Joab Jackson, Twitter Now Using Oauth authentication for Third Party Apps, Computer World UK, September 1, 2010, URL = http://www.computerworlduk.com/news/security/3237659/twitter-now-using-oauth- authentication-for-third-party-apps/

[25] Arne Roomann-Kurrik, Announcing gzip Compression for Streaming API's, Twitter Developers Feed, Jan 20, 2012, URL = https://dev.twitter.com/blog/announcing-gzip-compression-streaming-apis

[26] Aditya Mogadala and Vasudeva Varma. 2012. Twitter user behavior understanding with mood transition prediction. In Proceedings of the 2012 workshop on Data-driven user behavioral modelling and mining from social media (DUBMMSM '12). ACM, New York, NY, USA, 31-34.

[27] Johan Bollen and Huina Mao. 2011. Twitter Mood as a Stock Market Predictor. Computer 44, 10 (October 2011), 91-94. DOI=10.1109/MC.2011.323 http://dx.doi.org/10.1109/MC.2011.323

[28] Johan Bollen, Bruno Gonalves, Guangchen Ruan, and Huina Mao. 2011. Happiness is assortative in online social networks. Artif. Life 17, 3 (August 2011), 237-251.

[29] Manuel Cebrian. 2012. Using friends as sensors to detect planetary-scale contagious outbreaks. In Proceedings of the 1st international workshop on Multimodal crowd sensing (CrowdSens '12). ACM, New York, NY, USA, 15-16.

[30] Moheeb Abu Rajab, Lucas Ballard, Panayiotis Mavrommatis, Niels Provos, and Xin Zhao. 2010. The nocebo effect on the web: an analysis of fake anti-virus distribution. In Proceedings of the 3rd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more (LEET'10). USENIX Association, Berkeley, CA, USA, 3-3.

[31] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia. 2010. Who is tweeting on Twitter: human, bot, or cyborg?. In Proceedings of the 26th Annual Computer Security Applications Conference (ACSAC '10). ACM, New York, NY, USA, 21-30.

[32] Google. Google safe browsing API. http://code.google.com/apis/safebrowsing/, Accessed: Feb 5, 2010

[33] Jagadeesh Chandraiah. 2012. Fake anti-virus: The journey from Trojan to a persistent threat. Sophos Naked Security Blog

[34] Paul Ferguson. 2012. Observations on emerging threats. In Proceedings of the 5th USENIX conference on Large-Scale Exploits and Emergent Threats (LEET'12). USENIX Association, Berkeley, CA, USA, 4-4.

[35] Bo Han, Paul Cook, and Timothy Baldwin. 2013. Lexical normalization for social media text. ACM Trans. Intell. Syst. Technol. 4, 1, Article 5 (February 2013), 27 pages.

[36] Tyler Moore, Nektarios Leontiadis, and Nicolas Christin. 2011. Fashion crimes: trending-term exploitation on the web. In Proceedings of the 18th ACM conference on Computer and communications security (CCS '11). ACM, New York, NY, USA, 455-466.

[37] Chao Michael Zhang and Vern Paxson. 2011. Detecting and analyzing automated activity on twitter. In Proceedings of the 12th international conference on Passive and active measurement (PAM'11), Neil Spring and George F. Riley (Eds.). Springer-Verlag, Berlin, Heidelberg, 102-111.

[38] Donohue, Brian. 2013. Cool BlackHole Exploit Kits Created by Hacker. ThreatPots, Kapersky Lab Security News Service. January 9 2013.

[39] Howard, Fraser. 2013. Technical paper: Journey inside the Blackhole exploit kit. Naked Security from Sophos. November 30 2012

[40] C. E. Shannon. A Mathematical Theory of Communication, Reprinted with corrections from The Bell System Technical Journal, Vol. 27, pp. 379-423, 623-656, July, October, 1948.

[41] Twitter, "REST API v1 Resources" December, 2011, URL = https://dev.twitter.com/docs/streaming-api/concepts#sampling

[42] Bastian M., Heymann S., Jacomy M. (2009). Gephi: an open source software for exploring and manipulating networks. International AAAI Conference on Weblogs and Social Media.

[43] S. Martin, W. M. Brown, R. Klavans, and K. Boyack (to appear, 2011), OpenOrd: An Open-Source Toolbox for Large Graph Layout, SPIE Conference on Visualization and Data Analysis (VDA).