Intention and Origination: An Inside Look at Large-Scale Bot Queries

Junjie Zhang[*]
Wright State University
junjie.zhang@wright.edu

Yinglian Xie
Microsoft Research
yxie@microsoft.com

Fang Yu
Microsoft Research
fangyu@microsoft.com

David Soukal
Microsoft Corporation
dsoukal@microsoft.com

Wenke Lee
Georgia Institute of Technology
wenke@cc.gatech.edu

Abstract

Modern attackers increasingly exploit search engines as a vehicle to identify vulnerabilities and to gather information for launching new attacks. In this paper, we perform a large-scale quantitative analysis on bot queries received by the Bing search engine over month-long periods. Our analysis is based on an automated system, called SBotScope, that we develop to dissect large-scale bot queries. Specifically we answer questions of "what are the bot queries searching for?" and "who are submitting these queries?". Our study shows that 33% of bot queries are searching for vulnerabilities, followed by 11% harvesting user account information. In one of our 16-day datasets, we uncover 8.2 million hosts from botnets and 13,364 hosts from data centers submitting bot queries. To the best of our knowledge, our work is the first large-scale effort toward systematically understanding bot query intentions and the scales of the malicious attacks associated with them.

1 Introduction

As an indispensable service, search engines play an important role in our daily life. Unfortunately, a significant portion of the queries to search engines are bot queries that are generated by automated scripts and this volume continues to grow (at least 4.16% of search engine users are bots according to [17]). While previous work has focused on accurately detecting bot queries [14, 17], little attention has been devoted to understanding the nature of these queries. A natural question is "what are these bot queries searching for and who are submitting these queries?".

Motivated by recent observations that many bot queries are associated with malicious activities such as searching for vulnerabilities [20] or performing Search Engine Optimization (SEO) attacks [19], we perform a quantitative analysis of bot queries at a large scale. We believe that such analysis is important for understanding both the natures of bot queries and the scales of the related attacks. It can benefit many security applications. For example, precisely identifying vulnerability-searching queries provides valuable information on the newest security leaks and helps discover zero-day exploits [20, 22]. In addition, detecting hosts that constantly search trendy keywords or pharmaceutical topics provides opportunities to uncover blackhat SEO attacks that attempt to boost the ranks of malicious web sites [23]. Finally, detecting distributed, coordinated bot queries can help reveal the existence of botnets [11], their memberships, and their activity patterns.

As a means to this end, we describe a completely automated system called *SBotScope* for analyzing bot queries at a large scale. Different from previous work that focuses on specific types of malicious queries that are extracted based on domain knowledge or using seeds [20], our goal is to automatically dissect all bot queries in the following two aspects:

- *Query intentions:* Instead of investigating every single raw query, we look for an approach that can abstract *topics* that represent the underlying *query intentions*.
- *Query origination:* We are interested in analyzing hosts that submit bot queries, particularly those that have engaged in coordinated large-scale efforts. Many of these hosts are associated with botnet activities.

Despite extensive research on mining normal queries, analyzing bot queries has a number of unique obstacles. First, attackers adopt various strategies to obfuscate queries and to increase search coverage. For example, they often mix truly intended queries with random ones in a session, or combine stuffing words with the real relevant keywords to

[*]Work performed while at Microsoft Research Silicon Valley and the Georgia Institute of Technology.

diversify them. In addition, different from real user queries, bot queries usually do not result in any clicks as their primary purpose is to scrape information. Thus, we cannot leverage techniques that rely on clicks for understanding query intentions [8, 9, 6]. Finally, we are faced with a huge amount of data complexity, as the bot query volume is enormous and the query contents are highly diversified and constantly evolving.

To analyze bot query intentions, we develop a compact representation to summarize raw queries as a set of query patterns. This new representation forms the technical basis of our analysis. It allows us to hierarchically construct a set of topic trees that summarize query intentions syntactically and semantically. This approach captures the invariants of obfuscated queries, and further reduces the data complexity by orders of magnitude.

To understand the types of hosts submitting bot queries, we develop techniques to perform large-scale clustering of IP addresses and to identify distributed, coordinated query activities. We leverage topic trees to represent hosts as feature vectors and explore their temporal and spatial behaviors. Since existing work has mostly focused on detecting spamming botnets, we take a different angle to detect and analyze a large number of botnets that submit queries.

Putting things together, we make the following contributions in the paper:

1. Our work is the first large-scale effort towards automatically mining bot queries to understand query intentions and origins. We show that this direction of research can benefit security applications in multiple ways.

2. We develop a system called SBotScope to perform automated analysis and measurements. SBotScope can efficiently process billions of bot queries to categorize query topics and cluster hosts.

3. Applying SBotScope to two large query logs that were collected across different periods from the Bing search engine, we perform systematic analysis and present a set of unique findings.

Our study shows that a significant portion of bot queries can be attributed to a small number of malicious purposes despite a diverse set of query intentions among overall bot queries. In particular, 33% of bot queries are searching for vulnerabilities, followed by 11% attempting to harvest email addresses. The information collected by attackers via these queries can be used for zero-day exploits and spamming attacks. The remaining intentions range from content downloading, fashion items, news, tourism and geo-locations, to pornographic websites, and more.

We also demonstrate how we can leverage the analysis results of SBotScope to perform in-depth analysis of activities from both botnets and data centers. From our two datasets, we have identified 8,154,180 and 7,278,295 botnet IPs, and 13,364 and 19,559 data center IPs. Leveraging data center hosts to perform dedicated, malicious activities [1] is an emerging attack trend. Our analysis of coordinated search behaviors from data centers enables a unique perspective to quantitatively study this phenomenon at a large scale.

Finally, we show that our study can also help detect and mitigate malicious activities proactively. Using the generated query patterns, we detect tens of millions of additional bot queries that currently slip through the radar. These queries are likely submitted by stealthier hosts that mimic legitimate user behaviors.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 presents the system design, followed with experiment results in Section 4. Section 5 presents two example applications of SBotScope. Section 6 discusses attacker counter strategies and future work. We conclude in Section 7.

2 Related Work

There have been extensive studies on search queries for improving both search result rankings and user experience. To date, most research efforts in this field have focused on analyzing real user generated search queries (e.g., [16, 29, 21, 8, 9, 6, 27, 18]) so that we can better understand normal user intentions and improve query auto-suggestions.

While studying real user queries is important, we cannot ignore bot queries. Doing so may result in inaccurate statistics and polluted analysis results. Further, as the volume of bot queries continues to rise, they may deplete the resources of search engines, resulting in significant performance degradation for normal users. In this regard, previous work has proposed a few systems to more accurately detect bot queries [14, 15, 17]. For example, Buehrer et al. leveraged a supervised learning method [14, 15] and Kang et al. [17] adopted a semi-supervised learning method for detection.

Comparatively, little attention has been devoted to systematically understanding bot queries and studying their broad implications to security. Various recent evidences suggest that bot queries are often submitted from infected hosts from botnets [13]. They may also be submitted by attackers to collect information for a variety of malicious uses, such as targeted exploits, spamming, search engine optimization, and click frauds [26, 25, 7]. For example, a recent study shows that attackers submit queries to look for known vulnerabilities posted in underground web forums [20]. A separate study finds that attackers query trendy keywords in order to scrape contents from highly-ranked web sites and use them for promoting the rankings of malicious web sites [19].

With a massive number of sophisticated, actively evolving bot queries, a system that automatically summarizes and dissects bot queries can bring in a lot of value to security. Unfortunately, such a system is still in absence. As a result, the extent to which attackers have been exploiting search engine results remains largely unclear. Our work focuses on this new research problem, where we develop a system and perform strong quantitative analysis of bot queries to understand their intentions and the scales of malicious attacks associated with them on the Internet.

Compared with existing techniques on analyzing normal query intentions, our system differs in many important aspects. First, unlike [8, 16, 21], our system does not use click-through data since the majority of bot queries do not result in any clicks. Second, in contrast to [29, 27], our system does not depend on the availability of labeled data. Finally, different from [6, 9], our system does not need external sources of information, such as query-to-topic mapping functions based on Wikipedia.

Indeed, bot queries exhibit disparate characteristics compared to normal user queries, which make the approaches for analyzing normal queries not applicable to our study. First, both the labeled data and the query-to-topic mappings are relatively easier to obtain for normal queries than for bot queries. Compared with bot queries, the set of topic categories for normal queries are more stable, so they can often be derived manually. In addition, normal query intentions are more directly tied to the set of keywords in queries, as the purpose of normal users is to hit the relevant contents as fast as possible. In contrast, bot queries are more dynamic and their contents largely depend on the current security vulnerabilities and attack trends. The keywords in bot queries are more obscure in order to evade detection. They are also designed to increase the search result coverage with stuffing words. Such vast difference characteristics between normal and bot queries suggest that we cannot directly apply techniques for mining normal query intentions to analyze bot queries, but instead need to develop a customized system for our purpose.

3 The SBotScope System

Our goal is to analyze bot query intentions and origination by analyzing search logs automatically. Given the sheer volume of query records, manual investigation is infeasible. We need to develop a system to support such large-scale analysis on a regular basis.

Our system, SBotScope is designed to perform off-line analysis of search logs. It relies on existing bot query detection systems to generate bot query feeds for analysis. We assume that a large number of bot queries have been labeled by existing detection systems with a reasonable accuracy. Our system serves as a monitoring tool to discover query intentions, identify groups of bots performing coordinated search behaviors, and categorize queries accordingly. The output of SBotScope can be used in multiple ways. The categories and the statistics of query intentions can serve as input to security analysts for discovering new attack trends. The set of botnet and data center IP addresses can be used to block future attacks and to monitor attack scales. Finally, the output of SBotScope can also help detect additional bot queries that evade the existing detection systems.

Field	Description
IP	IP address
$Time$	Time stamp
$Query$	Query content
UA	Hash value of the User-Agent string
$Form$	Hash value of the query API
$Referrer$	Hash value of the Referrer string
$Click$	Whether any search result is clicked
$isBot$	Whether it is detected as a bot query

Table 1. Fields in a search query record

Data	Month	# of days	# of IPs	# of Bot Queries
D_1	May,2011	16	12,687,346	3,057,549,724
D_2	Oct,2011	16	12,207,937	3,198,810,465

Table 2. Datasets

In this section, we present the design of SBotScope in detail. We first describe our datasets, the challenges involved, and the system architecture. We then present the detailed data-processing flow.

3.1 Datasets

We obtain sampled data collected from the Bing search engine during two different periods. Table 1 shows the fields recorded for each query. Among these fields, the first seven fields are directly extracted from a query upon its arrival at the search engine. The last field "$isBot$" indicates whether this query is classified as a bot query by the current detection system, which mainly leverages an extensive set of features derived from query behaviors, e.g., the number of queries in a search session and the number of web pages visited from the user.

Table 2 shows the statistics of the datasets. Each dataset spans across roughly a two-week window, where we observe more than 3 billion detected bot queries, submitted from over 12 million unique IP addresses around the globe. The published privacy policies for this search engine address the storage, use, sharing, and retention of data collected in the course of the operation of these services. Our use of data is in compliance with these policies.

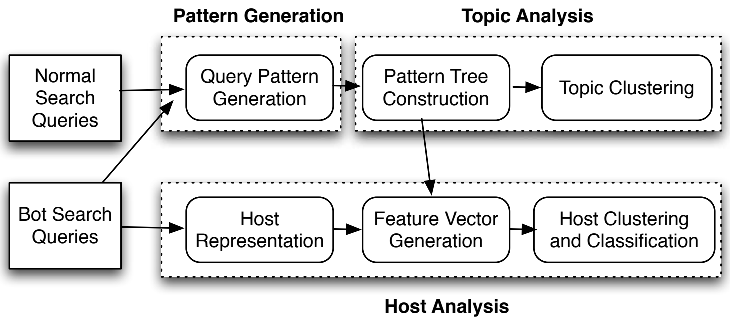

Figure 1. System Overview

3.2 Challenges and System Overview

The first challenging question of our analysis is how to represent query intentions. One popular approach, taken by normal query intention analysis [8, 9, 6], is to leverage the clicked search results. For example, given two different queries, if users always click the same set of result URLs, it is a strong indication that these two queries share similar intentions. However, such an approach is not applicable in our case because the vast majority of bot queries do not result in any clicks.

The exact raw query is another natural way to describe the underlying intention. This approach is also difficult as attackers often obfuscate queries or mix irrelevant queries with truly intended ones. Table 3 shows an example of a bot-query session [1]. In this case, attackers not only add stuffing words, e.g., “unled” and “cheap mini”, to increase their query diversity, but also add popular queries such as “facebook.com” and “bing.com” to make their query session look more legitimate. The real intentions of these queries are looking for vulnerable web sites powered by certain versions of software.

Intuitively, to capture intentions, we would like to derive the invariant portions of the queries with truly relevant keywords and randomly inserted stuffing words. Toward this goal, we derive a set of query patterns in terms of invariant keywords to represent raw queries in a more compact format (see Section 3.3). The use of patterns also dramatically reduces the complexity of our analysis. Furthermore, we aggregate query patterns into a small set of topic trees that summarize query intentions both *syntactically* and *semantically*. This step allows human users or security analysts to conveniently examine the composition of queries.

The second question is how to represent hosts and identify their correlated activities, especially when the volume of query records is huge. As Table 2 shows, each of our datasets contains billions of queries and the total volume of data is on the order of terabytes. Therefore, our system needs to effectively reduce data complexity and filter noise to study query origins.

[1]We use the term “session” to loosely refer to a set of consecutive queries issued from one IP address within a short duration.

unled “powered vbulletin version 3.8.6”
cheap mini “powered vbulletin version 3.8.4”
discussion “powered vbulletin version 3.8.7”
facebook.com
forum “powered vbulletin remove”
bing.com
email “powered vbulletin music”

Table 3. An example bot-query session

To address this challenge, we construct feature vectors that represent hosts in a low dimensional subspace. It further aggregates hosts that share similar query patterns into clusters for analysis. Specifically, we implement a parallel version of the single-linkage hierarchical clustering algorithm [24] that enables us to perform host clustering in a fully distributed fashion. SBotScope then classifies host clusters based on their temporal and spatial behaviors. This approach allows us to perform in-depth analysis of queries submitted from botnets and data centers separately.

To summarize, Figure 1 shows the architecture overview of the analysis performed by SBotScope. We describe the details next.

3.3 Pattern Generation

We would like query patterns to capture only the invariant portions of queries and remove irrelevant stuffing words. In doing so, it is important to avoid extracting popular combinations of stopwords (e.g., “of”, “the”) as well as common, popular queries such as “facebook.com”.

To capture the above intuition, we define a *query pattern* as a *specific word-combination (or query)* that occurs *frequently* among bot queries. By *“specific”*, we require the pattern to contain useful information, so that to avoid selecting over-general patterns, such as those of stopwords.

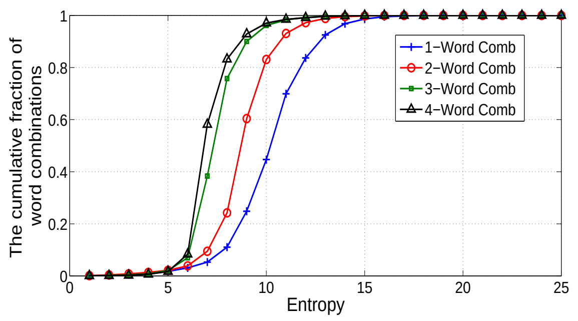

Figure 2. The distribution of entropy for frequent word-combinations with certain length

By *frequent*, we require a pattern to be the common, invariant portion of a large number of queries. Finally, we consider word-combinations instead of query sub-strings or regular expressions to prevent attackers from manipulating word orders or query formats to defeat our analysis.

3.3.1 Selecting Frequent Patterns

The "frequent" notion can be naturally quantified using the pattern frequency. Without loss of generality, we consider both frequent word combinations and frequent exact queries. Since we find that majority (95%) of the bot queries contain at most four words, we consider all possible word combinations up to length 4 for each query. Then for every length (i.e., 1, 2, 3, and 4), we select the top 0.1% combinations that occurred across the most number of unique queries. After this process, there are still some frequent exact queries that are not covered by popular word combinations. For such cases, we also select frequent exact queries (i.e., top 0.1%) as potential patterns.

3.3.2 Selecting Specific Patterns

Once we identify frequent patterns, we proceed to select specific ones using the following two metrics:

- *Information gain* $G(w_i)$ quantifies the amount of information contained in a pattern w_i:

$$G(w_i) = E(Q) - E(Q|W = w_i)$$

  where $E(Q)$ is the entropy for all bot queries, and $E(Q|W = w_i)$ is the conditional entropy of bot queries matching pattern w_i. Since $E(Q)$ is a constant for the same set of queries Q, $G(w_i)$ can simply be evaluated by $E(Q|W = w_i)$. The smaller the conditional entropy, the larger the information gain is, and hence the corresponding pattern is more specific. In the exact query case, the pattern is the entire query itself, so the information gain is the maximum value of $E(Q)$.

- *Historic popularity among normal queries* $c_{normal}(w_i)$: it refers to the number of unique IP addresses that submitted normal queries matching pattern w_i. The intuition is that a very specific bot query is less likely to be a popular query among a large number of normal users. A higher value of $c_{normal}(w_i)$, means the pattern is not distinguishing; we find that many of them are used for obfuscating sessions or testing connectivity (e.g., "facebook.com").

Figure 2 presents the conditional entropy $E(Q|W = w_i)$ distribution for frequent patterns of different lengths (we exclude exact query patterns). As shown in the figure, patterns with more words tend to have smaller conditional entropy, since they are naturally more specific (i.e., containing more words). Based on these two metrics, SBotScope selects specific patterns if w_i satisfies either of the following conditions:

- $E(Q|W = w_i)$ falls in the bottom 90% of entropy for all popular word combinations with the same length.

- $c_{normal}(w_i)$ is smaller than a certain threshold $1,000$.

We use these two conditions to eliminate over general patterns or patterns that are popular among normal user queries.

Note that given a set of patterns, a query may match multiple of them and contribute to all their frequencies. For each query, to avoid generating multiple redundant patterns, we select only the most specific pattern to represent it (i.e., we favor the pattern that has the smallest conditional entropy). After the above pattern generation process, we can dramatically reduce the data complexity by several orders of magnitude, from 10^9 queries to 10^5 patterns.

3.4 Topic Analysis

The topic analysis step is where we further analyze query patterns to identify semantically meaningful topics. Patterns represent the prominent syntactic features of raw queries, but different patterns may still correspond to the same or similar intentions. For example, "auto" and "car" are very different words, but they all correspond to similar topics and our goal is to uncover the hidden semantic correlations among them.

One potential approach to recognize semantically similar patterns is to use dictionaries. However, as bot queries are written in various languages, it is challenging to parse each

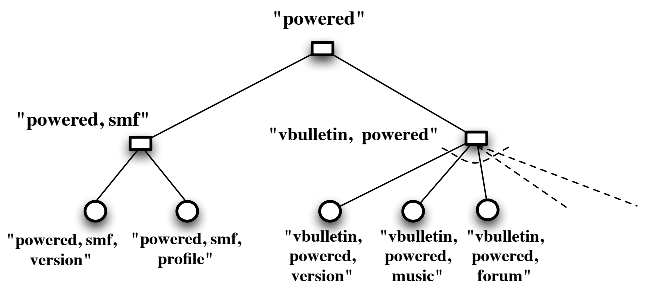

Figure 3. An example pattern tree

query and understand it semantically. In our work, we adopt a simple yet effective process that includes two steps. The first step aggregates patterns hierarchically to construct a set of pattern trees. The goal is to further reduce the pattern complexity syntactically. The second step performs clustering on the pattern trees to identify semantically coherent topics.

3.4.1 Pattern Tree Generation

This step aggregates patterns based on shared common words. We take a bottom-up approach and treat each input pattern as an individual node. The tree construction starts with merging the longest patterns with length K if they share $K-1$ common words. The merged pattern is represented as a new parent node of the corresponding leaf pattern nodes. This process repeats until all the top-level tree nodes represent 1-word patterns and thus cannot be further aggregated. Figure 3 shows an example pattern tree constructed using this process. All the patterns on this tree share the common word "powered", and they may all be issued to discover certain vulnerabilities. In particular, lower level siblings are more strongly correlated, as they share more common words than nodes in higher levels. In our experience, this step can further reduce hundreds of thousands of patterns to roughly two or three thousand pattern trees.

3.4.2 Topic Clustering

This step groups pattern trees to identify their semantic correlations. We use the probability of word co-occurrence to model such correlations. For example, the word "auto" and "car" may both co-occur with keywords such as "Toyota" or "dealership", which can link the two concepts together into one group.

Specifically, we focus on only the top-level nodes. Given two top-level pattern tree nodes with word w_1 and w_2, we use a similarity score $s(w_1, w_2)$ to measure their correlation. We first identify the two sets of queries Q_1 and Q_2 that contain word w_1 and w_2, respectively. We then compute the similarity score in terms of the Jaccard distance between Q_1 and Q_2:

$$s(w_1, w_2) = \frac{|Q_1 \cap Q_2|}{|Q_1 \cup Q_2|}$$

With the similarity scores, we essentially obtain a graph of pattern trees, with each tree becoming a super node in the graph. The similarity scores serve as edge weights. Given the graph is reasonably small (with a few thousands of nodes), we perform spectral clustering [12], the best graph-cut algorithm, to identify tightly connected subgraphs. Each subgraph intuitively corresponds to a query topic.

The number of topics can be a pre-configured parameter to control the granularity of topics (i.e., coarse-grained categorizations vs. detailed classification). In our current system, we pick the number of topics to be 50, which allows security analysts or search engine operators to manually inspect the results.

Data	# of IP-Segments
D_1	14,856,347
D_2	14,350,435

Table 4. The number of "hosts" in two different data sets.

3.5 Host Analysis

The host analysis step answers the question of "who submitted bot queries?". We are interested in identifying and separating data center hosts from botnet hosts, which we find are the two dominant cases. The infrastructures of these two categories of hosts are drastically different, so their query behaviors and contents may also differ radically. Such a study could yield in-depth understanding to attacker strategies and attack scales. In our analysis, we group hosts with similar query behaviors via clustering.

3.5.1 Host Representation

With DHCP, a host's IP address may dynamically change. To mitigate this problem, we use an *IP-segment* to represent a host that has been active during a period. We find operating on the coarse-grained time period on the order of days generates stable results. For example, if we observe an IP address that submitted bot queries on day$_1$, day$_2$, day$_3$, and day$_5$, then we use two IP-segments to represent the two "hosts" that issued these queries: IP-segment$_1$ $=\{$IP: day$_1$, day$_2$, day$_3\}$ and IP-segment$_2$ $=\{$IP: day$_5\}$. We apply this method to the two data sets to obtain hosts. The number of hosts are similar across two data sets as presented in Table 4. Furthermore, the number of such defined "hosts" is

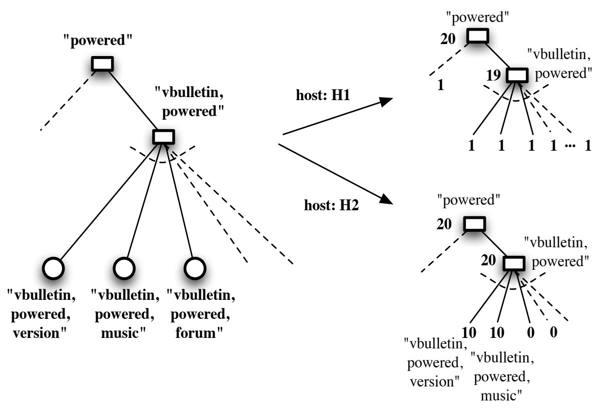
"powered"
"vbulletin, powered"
"vbulletin, powered, version"
"vbulletin, powered, music"
"vbulletin, powered, forum"
host: H1
host: H2
20
19
1
1 1 1 1 ... 1
20
20
10 10 0 0

Figure 4. An example of generating feature vectors

close to the number of unique IP addresses in Table 2, which implies that the vast majority of the IP addresses are either active on one day or active on consecutive days.

3.5.2 Feature Vector Generation

With the definition of “hosts”, we construct a feature vector to represent each host. One straightforward solution is to define the feature vectors in the original query space, where each unique raw query is one dimension. Given that the sheer volume of unique bot queries is huge, this option will be prohibitively expensive in practice.

Instead, we leverage the derived pattern trees, which is much more compact than raw queries. For each host H represented by an IP-segment, we match all of its bot queries against the patterns on the trees starting from the root nodes, using the following steps:

1. SBotScope first produces a set of subtrees $T(H)$ where all the patterns on the subtrees match the queries from host H.

2. For each node n on $T(H)$, SBotScope additionally records the number of queries from H matching n as $c(n)$.

3. SBotScope then prunes $T(H)$. For an edge $a \to b$ on $T(H)$, if $c(a) >> c(b)$ (we consider $c(a) >> c(b)$ if $log_{10}(c(a)) >= log_{10}(c(b)) + 1$), then a is a representative pattern for H while b is not. So we remove the subtree rooted at b.

Finally, we pick the set of leaf nodes on the pruned subtrees to serve as features for describing H. The total number of dimensions in the feature vector space is the number of unique pattern nodes selected across all hosts. Figure 4 presents an example. In this example, H_1 generates 20 queries and has matching count $c(\text{“powered”}) = 20$ and $c(\text{“vbulletin, powered”}) = 19$. The remaining nodes match only one query from H_1. Therefore, we pick “vbulletin, powered” as a feature to describe H_1. For another host H_2, we pick “vbulletin, powered, version” and “vbulletin, powered, music” as two features to represent H_2.

This step is critical as it significantly reduces the feature vector dimensions from billions of raw queries to only tens of thousands of features (e.g., 27,507 for D_1 and 37,843 for D_2). More importantly, the feature vectors describe not only the distinguishing local query contents of each host, but also the global correlations among different query patterns.

3.5.3 Host Clustering

With a feature vector to represent a host, we can group hosts into clusters and classify their types based on the aggregated behaviors of each group. This approach has two advantages than directly analyzing individual hosts. First, it allows us to identify correlated or coordinated query behaviors that cannot be observed in isolation. Second, it is robust to individual host outliers as we rely on group behaviors rather than host behaviors to perform classification. Thus, we are able to differentiate hosts even though some of their activities are atypical and are difficult to classify otherwise.

To perform host clustering, we apply the single-linkage hierarchical clustering algorithm [24] using the Jaccard distance of feature vectors:

$$dist(H_1, H_2) = 1 - \frac{|F_1 \cap F_2|}{|F_1 \cup F_2|}$$

where F_1 and F_2 represent the set of features for H_1 and H_2 respectively. If the Jaccard distance of two hosts is smaller than a pre-defined threshold (e.g., 0.3), we aggregate two hosts into one cluster. We choose the single-linkage hierarchical clustering algorithm instead of the popular K-Means algorithm [10] because it does not need to predetermine the number of clusters beforehand. In addition, it enables parallel implementation in a fully distributed fashion on a computer cluster. In the current implementation, we keep a cluster if it includes at least 10 hosts.

3.5.4 Cluster Classification

To answer the question of what types of hosts are from these clusters, we classify the group behaviors. Two distinguishing categories of hosts are botnets and data center hosts. A botnet is a collection of compromised machines performing malicious activities. Identifying botnets helps categorize queries with malicious intentions. The behaviors of botnet hosts are usually bursty in time [30]. Their IP addresses are widely distributed across a large number of networks. In contrast, data center hosts are long-lived, well-maintained machines for dedicated tasks. Their IP addresses usually belong to a small number of administrative domains.

We explore both the temporal and spatial characteristics of hosts to classify a cluster. The temporal behaviors describe the persistence of the automated search behaviors, while the spatial behaviors describe the distribution of host IP addresses across the Internet. To quantify the spatial behaviors of a cluster, we use the ratio of the number of unique /24 prefixes over the number of unique IP addresses in a cluster, defined as r. Specifically, $r = \frac{num\ of\ /24\ prefixes}{num\ of\ IPs}$. We use the following two criteria in our classification:

1. If the majority (90%) of hosts in a cluster are active for only a short period ($\leq$ 3 days) and the host IP addresses are widely distributed ($r \geq 0.7$), we classify this cluster as a likely botnet cluster.

2. If the majority (90%) of hosts in a cluster are long-lasting ($\geq$ 10 days) and the IP addresses are from a small number of different networks ($r \leq 0.3$), we classify the cluster as likely a data-center cluster.

This step cannot classify all host clusters, but it does identify groups with well-known, distinguishing behavior patterns. Only a small percentage of clusters (around 4%) cannot be classified using our approach.

4 Measurement Results

We apply SBotScope to the two datasets described in Table 2 (see Section 3). We present our experiment results in this section.

4.1 Query Intention Analysis

Datasets	Query Patterns (word-combinations)	Query Patterns (exact queries)
D_1	361,568	364,405
D_2	370,338	593,349

Table 5. The number of patterns derived from each dataset.

The *Pattern Generation* process is our technical basis for deriving query intentions. Table 5 shows that this step reduces the data from over 3×10^9 raw queries to fewer than 10^6 query patterns. These patterns can be further aggregated into 1,823 and 3,499 pattern trees for dataset D_1 and D_2, respectively.

After further grouping pattern trees into semantically correlated topics, we examine the compositions of query intentions in detail. Table 6 presents the top 6 most popular topics as well as the top 5 query patterns that match the most number of queries under each topic. Previous work has suggested the use of bot queries for searching for vulnerabilities. Indeed, we find vulnerability-searching queries are the dominant category, corresponding to almost 1/3 of all bot queries. The second most popular topic (around 11% of bot queries) is about searching for email addresses or user accounts, which include critical information for attackers to perform spamming or account hijacking attacks more effectively. The next four popular categories are used for content downloads, fashion items (e.g., handbags, cloths, and watches), car sales, and news.

Investigating further into the vulnerability searching queries, we find that this topic includes 170 pattern trees and 7,295 query patterns. Figure 5 shows three example pattern trees under this topic. A majority of query patterns in this category contain keywords that are known to be related to vulnerabilities such as “php”, “yabb”, “powered”, “topic”, “forum”, “thread”, “board”, and “vbulletin” [20]. In particular, both “powered” and “php” are popular words with a large number of subtrees under each of them. These examples demonstrate the effectiveness of SBotScope in identifying semantically correlated patterns even when they are not strongly connected syntactically (i.e., they do not share common words). These vulnerability-searching queries provide information to prevent future attacks. For example, Table 7 shows three known vulnerabilities [2] that one can derive from our query patterns, and we leave it as future work to further explore this direction.

Query Patterns	Vulnerability
powered by photo album	PHP Album 0.3.2.3 Remote Command Execution
powered by update	PHP-Update 2.7 Remote Code Execution
powered by icalendar	PHP iCalendar 2.24 File Upload

Table 7. Example vulnerabilities derived from query patterns

Data	# of clusters	% of queries	% of clusters validated
D_1	39,037	90.30%	82.96%
D_2	47,217	82.18%	85.89%

Table 8. Clustering Results

4.2 Query Origin Analysis

We now examine the query origin distributions and classification. After performing the host clustering analysis described in Section 3.5, we obtain in total 39,037 and 47,217 clusters for dataset D_1 and D_2, respectively. As indicated in Table 8, these clusters contain 90% and 82% of all bot queries, suggesting that a vast majority of bot queries are generated from coordinated hosts.

Topic	% of Queries	Top 5 Patterns	Query Samples
Vulnerability Discovery	32.8%	list members mode php mode php register es php page aspx html php powered by	debate members mode list php ucp php mode register user php page es X-Powered-By aspx html php powered by php register
Email Harvest	11%	yahoo.cn email 163.com email 21cn.com email sina.cn email 163.net email	beijing bank email yahoo.cn beijing food email 163.com International email 21cn.com forum email sina.cn bbs email 163.net
Content Download	3.6%	download free flash free games online coupons online games play movie trailer	free flash download 2010 free games online adult coupons online imax california play games sony psp movie trailer spring 2011
Fashion Items	1.4%	replica handbags designer handbags Black Footwear Leather clothing store buy digital watches	CA replica handbags cheap designer online handbags Nike Black Footwear Leather mini lady clothing deal store buy 2010 digital watches
Car Sale	1.3%	accord honda used dealer used ford dealer used mercedes dealerships used hyundai dealerships used vw	accord honda used GA dealer used ford USA NY number dealer used mercedes 98052 dealerships zipcode used hyundai dealerships used vw online
News	0.7%	fox live news 10 channel news dallas morning news 2011 latest news celebrity gossip	fox live news 2011 10 channel news 2010 local dallas morning news april 2011 2011 latest news archive 2010 celebrity gossip photos jpg

Table 6. Top six popular topics and their top five patterns

We further classify the group temporal and spatial behaviors to separate botnet clusters and data center clusters. We combine all the clusters from dataset D_1 and D_2, and plot each cluster as a point in a two-dimensional space in Figure 6.

The X-axis represents the ratio of the number of /24 network prefixes over the number of unique IP addresses in a cluster, i.e., r in Section 3.5.4. A small ratio means the majority of hosts in a cluster come from a small number of network ranges, so the cluster is more likely to consist of hosts from common administrations. In contrast, a large ratio in the X-axis suggests that the cluster is more widely dispersed in the Internet, with each network range having only a few hosts.

The Y-axis represents the average active periods of a host (in terms of days) in a cluster. Long-lived clusters are more likely data center hosts, while transient clusters are more likely associated with botnet activities.

From the figure, we observe that the majority of the clusters are transient and have widely distributed IP addresses, indicating the existence of a large number of botnets. The density of the points in the figure drops significantly as the ratio r decreases and the average number of active days increases. We find that the number of data center clusters is relatively small. Applying the heuristics in Section 3.5.4, we identify in total 37,268 and 44,252 botnet clusters, and 137 and 150 data center clusters from D_1 and D_2, respectively. Accordingly, we obtain 8,154,180 and 7,278,295 IPs in botnet clusters, and 13,364 and 19,559 IPs in data center

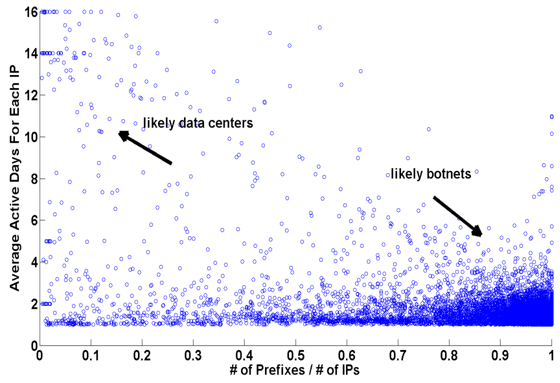

Figure 6. "$avg\ active\ days\ per\ IP$" **v.s.** "$\frac{\#of\ /24\ prefixes}{\#ofIPs}$" **in one cluster**

clusters from D_1 and D_2. Only a small percentage of clusters, 4.2% in D_1 and 6% in D_2, remain unclassified, and we leave them for future studies.

The existence of botnet activities is perhaps not surprising [13], but the prevalence and scale of using botnets for submitting bot queries are more than we had expected. These query-submitting botnets represent an important category of botnets that are not well studied yet. Their query contents and behavior patterns will be valuable to analyze the trends and scales of malicious activities.

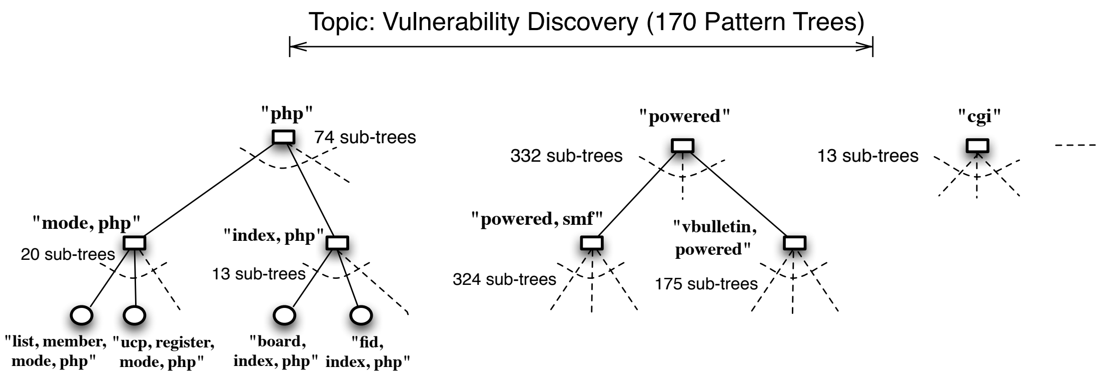

Figure 5. Three example pattern trees for searching vulnerabilities.

The number of data center clusters is also non-trivial. Apart from legitimate activities from data centers, the existence of malicious activities from data centers may indicate a new trend, where attackers have started exploiting cloud-computing or other well-maintained infrastructures for launching attacks.

4.3 Cluster Validation

The quality of the identified clusters and their classification could be a concern due to the unsupervised nature of our clustering process. Ideally, we would like to obtain the ground-truth information regarding botnet memberships and data center usages. Nevertheless, it is extremely hard, if not impossible, to build such ground-truth in practice.

In the lack of ground-truth, we adopt the following three methods to sanity check our results. First, we evaluate the quality of clustering using the extra fields presented in Table 1. Intuitively, if a host group was correctly identified, they may exhibit similar behaviors in choosing fields such as UA, $Form$, $Referrer$, as they all use the same script. Second, we perform Whois [5] lookups to examine the names and the types of the ASes that correspond to different types of clusters. The purpose is to validate whether botnet hosts indeed mostly correspond to consumer networks, and whether data center hosts indeed belong to well known data center networks or hosting services. Third, we compare our botnet host lists with those derived from the Conficker IP address lists in [28] by their courtesy.

4.3.1 Cluster Behavior Similarity Analysis

For each cluster, we examine whether the majority of its "hosts" have identical field values. For each host H, we first pick its dominant values for UA, $Form$, and $Referrer$. We then consider these metrics for both the queries matching patterns as well as the queries not matching any patterns.

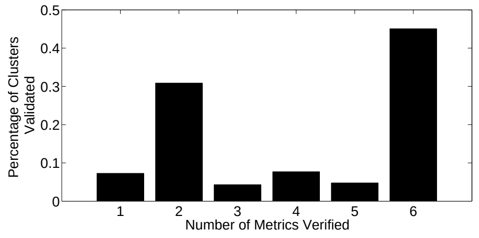

Figure 7. Percentage of clusters sharing identical metrics.

In total, we have six metrics. We then compute the percentage of a cluster sharing identical values for these metrics. For each metric, we also consider the case where each host randomly chooses a different field value (e.g., by adopting randomization strategies to evade detection). In this case, we set the value of the corresponding field to "random" if its values exhibit *complete* randomness.

Figure 7 shows the distribution in terms of the number of metrics on which a cluster shows strong consistency (i.e., 90% of hosts in this cluster share an identical value). Overall, 82.96% clusters from D_1 and 85.89% clusters from D_2 share at least one identical metric. A large percentage of clusters have all six values identical across hosts. These are strong indications of coordinated behaviors.

4.3.2 Host Network Type Analysis

As botnet and data center clusters usually come from different network regions, we reverse lookup the autonomous system number (i.e., AS number) for each IP address. In addition, we perform Whois [5] lookups on the AS numbers, and examine the names and the types of networks.

Table 9 lists the top five most popular AS names for both types of clusters. We observe that botnet clusters include mainly hosts from residential broadband ISPs and consumer

botnet type		
ASN	Country	AS Name
4134	CN	CHINANETBACKBONE
8151	MX	Uninet S.A. de C.V.
3269	IT	ASN-IBSNAZ
27699	BR	TELECOMUNICACOES
4837	CN	CHINA169-BACKBONE
data center type		
ASN	Country	AS Name
15003	US	NOBISTECH
36351	US	SOFTLAYER
25973	US	GTT
21788	US	NOC
13647	US	TRANQUIL-HOSTING

Table 9. Top 5 ASes for both types of clusters

IP/Prefix	% in the Conficker Botnet
IP	11.29%
/24 Prefix	72.75%

Table 10. Percentages of botnet IPs/prefixs in the top 5 ASes that appear in the Conficker botnet

access networks, while data center clusters mainly consist of hosts from cloud computing or hosting service infrastructures. For example, NOBISTECH [3] and SOFTLAYER [4] are two large service providers that provide hosting services and data center services.

4.3.3 Comparison with Known Botnet Hosts

We also compare our botnet IP address list with the set of known Conflicker bots derived using methods from [28] by their courtesy. We find that all the top five popular ASes in our botnet clusters are the exact same set of popular ASes for Conficker bots. We also compare the IP addresses from the top 5 ASes (see Table 9) in our botnet clusters to the known Conficker IP address set. Table 10 shows that 11.29% of the botnet IPs identified by SBotScope overlap with the set of known Conficker IPs, and 72.75% of the botnet IPs fall into the set of known /24 Conficker IP prefixes. This result suggests that the subset of botnet hosts that overlap with Conflicker bots are indeed correctly identified and classified.

4.4 Comparing Botnet and Data Center Query Activities

The categorization of query contents and host types allows us to perform comprehensive analysis and comparisons on the query intentions and the network infrastructures from botnets and data centers. In this section, we quantitatively study the different behaviors and intentions from these two types of hosts.

	Botnet clusters	Data center clusters
D_1	php	black
	download	light
	powered	gold
	email	video
	pdf	bank
D_2	email	certified
	php	flights
	topic	car
	download	shirt
	blog	adult

Table 11. Popular keywords in the patterns

4.4.1 Comparison on Search Behaviors

We find that the search contents and behaviors of the two types of clusters are drastically different. Table 11 summarizes the most popular words in the query patterns from different types of clusters. We find that botnet clusters mainly focus on discovering vulnerabilities (e.g., using word “php”, “powered”, “pdf”), searching for email addresses and user account information (e.g., “email”), and looking for free content downloads (e.g., “download”). In contrast, data center clusters focus more on looking for commercially relevant information using keywords related to apparel and entertainment.

	botnet type	data center type
# of clusters	37,268	137
# of IP-segments	8,355,098	13,853
# of IPs	8,154,180	13,364
avg # of IPs/cluster	219	98
# of queries	2,278,143,556	101,638,148
avg # of queries/IP	279	7,605
avg # of patterns/cluster	148	4,208

Table 12. Host activity summary for dataset D_1

	botnet type	data center type
# of clusters	44,252	150
# of IP-segments	7,423,261	20,099
# of IPs	7,278,295	19,559
avg # of IPs/cluster	164	130
# of queries	1,444,931,738	198,903,861
avg # of queries/IP	198	10,169
avg # of patterns/cluster	166	3,121

Table 13. Host activity summary for dataset D_2

Besides different query contents, we also observe distinguishing activity patterns. Table 12 and 13 summarize the statistics of the host activities from dataset D_1 and D_2. We find that the results are consistent over time. Overall, botnet clusters include more IP addresses and submit an order of magnitude of more queries in total than data center clusters. In contrast, data center hosts are usually long-lived; each host submits more queries on average, with a larger number of query patterns per cluster.

4.4.2 Comparisons on Network Infrastructures

We then analyze the network infrastructures used by the two different types of clusters. The number of botnet clusters clearly dominates among the results. SBotScope identifies 37,268 botnet clusters and only 137 data center clusters from dataset D_1. Similarly, it identifies 44,252 botnet clusters and 150 data center clusters from dataset D_2.

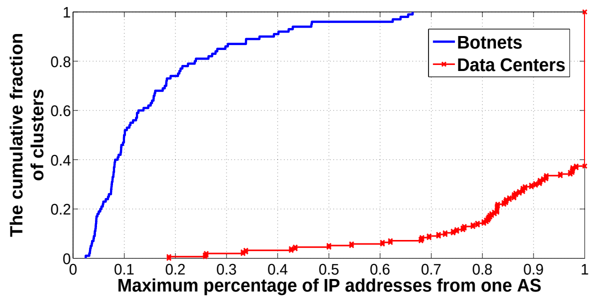

Figure 8. Distribution of the maximum percentage of IP addresses from one AS for each cluster

We find that hosts in a data center cluster are more likely to be from the same organization, perhaps due to the lower costs and efforts in comparison to the case of renting hosts from many organizations. We reverse look up the AS number for each IP address in data centers clusters. We randomly pick 1000 botnet clusters (since the number of botnet clusters is significantly more than the number of data center clusters) and look up the AS numbers for all of their hosts. For each cluster, we identify the AS with the most number of hosts and compute the percentage. Figure 8 presents the corresponding distribution. More than 90% data center clusters have a large fraction of hosts ($\geq$ 75%) hosts coming from the same AS. In contrast, less than 5% botnet clusters have a dominant AS with over 50% of their hosts.

	D_1	$D_1 \cap D_2$	% of persistent IPs
Botnet IPs	8,154,180	489,251	6%
Data center IPs	13,364	4,544	34%

Table 14. Host overlap across time

In addition, we also compare the set of classified IP addresses across the two datasets collected from different periods. We find that data center hosts are much more stable over time. As presented in Table 14, for data center IP addresses that appeared in May, 2011 (D_1), 34% of them were still active and also belong to the data center clusters in October, 2011 (D_2). As a comparison, only 6% of the botnet IP addresses in May, 2011 overlap with the botnet IP addresses in October, 2011.

To summarize, our findings indicate that botnet is a popular means for attackers to gather vulnerabilities and other information from search engines. The majority of the bot queries come from botnets on a global scale. Bot queries from data centers are of mixed categories. We observe both queries that are triggered by real users as well as malicious queries. The appearance of data center hosts performing malicious activities is a relatively new trend and deserves futher study.

5 Case Studies and Applications

In this section, we perform in-depth analysis on the intentions and the origins of botnets and data center clusters using three case studies. From these case studies, we show how one can derive different attack phases, identify the precise botnet memberships, and uncover the exact vulnerabilities targeted by attackers.

One application of SBotScope is to leverage the derived query patterns for detecting bot queries that slipped through the radar. To our knowledge, all existing bot-query detection systems rely on host (or user) behaviors or IP address information for detection. None of them have explored the use of query contents, which can be more robust to attacker counter strategies, as it is fundamentally difficult to modify query keywords without affecting search results.

5.1 Case Study I: A Botnet Cluster

One detected botnet cluster is distinguished by the query pattern "*WordPress forum plugin Fredrik*". Table 15 shows the statistics for this cluster. It includes 1,807 IP addresses distributed across 1,752 /24 network prefixes, submitting in total 127,557 queries with 62,115 unique ones. The vast majority of hosts are active for only 1 day.

# of IPs	1,807
# of /24 prefixes	1,752
# of queries	127,557
# of unique queries	62,115

Table 15. Statistics of a botnet cluster

Tokens for vulnerabilities	# of queries
WordPress forum plugin by Fredrik Fahlstad	48,010
fbconnec_action=myhome	10,699
plugins/wpforum	1,160
forum.php	1,150
Powered by Joomla!. valid XHTML and CSS	1,012

Table 16. Most popular vulnerability tokens in the queries.

We examine this cluster and find that a large fraction of its queries are related with vulnerability searching. Each query contains a quoted sub-string token for discovering different types of vulnerabilities. There are in total 443 unique sub-string tokens. Table 16 lists the top popular tokens that were searched most frequently. In addition to the quoted sub-string tokens, each query also contains additional words, either specifying the query scopes, or aiming to increase the search coverage.

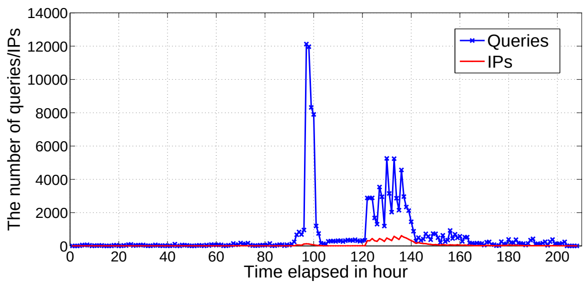

Figure 9. Number of queries/IPs in each hour

This botnet cluster shows strong bursty patterns in its temporal behavior. Figure 9 plots the number of queries and the number of IP addresses observed overtime. We see two clear bursty periods, one for 6 hours, and the other for 21 hours. A closer look at these two bursts (Table 17 and 18) reveals a multi-phase strategy adopted by attackers.

Burst1	Duration (in hours)	6
	# of IPs	149
	# of queries	41,276
Burst2	Duration (in hours)	21
	# of IPs	1,682
	# of queries	53,821

Table 17. Detailed statistics of the two phases

Burst1	"index.php/thread-" shooter
	"yabb/yabb.pl?board" siver
	"index.php/thread-" site:com
	"register.php?do=" socal
	"memberlist.php?page=" somerset
	"active_topics.asp?at=" sorrowing
	act "WordPress forum plugin by Fredrik Fahlstad" site:.edu
Burst2	inch "WordPress forum plugin by Fredrik Fahlstad" site:.cn
	plus "WordPress forum plugin by Fredrik Fahlstad" site:.net
	change "WordPress forum plugin by Fredrik Fahlstad" site:.com

Table 18. Example queries submitted in the two phases

In the first burst, a relatively small number (149) of hosts are involved, but they are very aggressive in generating queries (submitted 41,276 queries in merely 6 hours). The average number of queries per IP address is 277, and the average interval between two consecutive queries is only 4.6 seconds. Most of the bot queries are in the format of a quoted vulnerability token, followed with a random string or word for increasing the query diversity. There are a large number of different vulnerability tokens queried in this period. The first row in Table 18 lists some examples.

The second burst involves more bots in this cluster, with 1,682 out of 1,807 hosts active. There are 49 bots in the first phase also active in the second phase. These bots are more persistent (e.g., active for 21 hours compared to 6 hours in the first phase), and their search behaviors are much more stealthy. Each bot generated fewer queries (31 queries on average) and the average interval between two consecutive queries is 21 seconds, significantly higher than the 4.6 seconds before. The second row in Table 18 shows the example queries submitted in this stage. These queries all target at a specific type of vulnerability with different site scopes.

Considering the different but correlated query strategies used in these two bursts, we suspect that queries in the first burst were mainly used to explore the diversity of the vulnerable web sites, while those in the second burst focused more on finding information about a specific vulnerability, which could be the most popular one or the easiest one for exploitation. This example provides us with detailed understanding into the sophisticated query strategies by attackers. Our system can effectively aggregate such activities and provide important information for attack detection and defense.

5.2 Case Study II: A Data Center Cluster

# of IPs	446
# of /24 prefixes	29
# of queries	873,064
# of unique queries	495,211
# of names	354,709

Table 19. A case study of data center cluster

One of the interesting data center clusters that we identify share the query pattern *"-Genealogy, -Generation"*. Table 19 presents the cluster statistics. It has 446 distinct IP addresses coming from 29 /24 IP prefixes. In addition, all the IP addresses have been actively generating bot queries throughout the data collection period. Despite the fact that the 29 /24 IP prefixes look quite different, reverse DNS lookups show that all of the IP addresses actually belong to the same organization "SOFTLAYER", which appears to be an hosting service.

This cluster generates in total 873,064 queries with 495,211 unique ones, looking for information using people names, possibly for background investigation. Each bot first initiates a query with only a name (e.g., *"alice foo"*), immediately followed by another query with the same name plus additional constraints (e.g., *"alice foo" +("alice is" OR* *"alice was") -Genealogy -Generation language:en*). We see 354,709 unique names being queried.

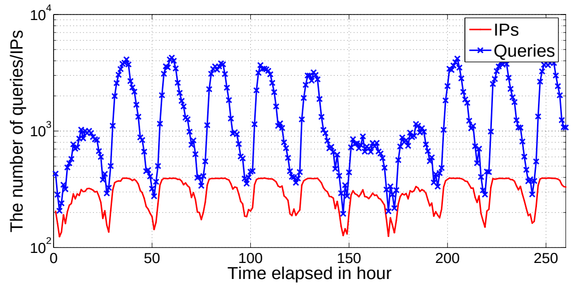

Figure 10. Number of queries/IPs in each hour

Figure 10 shows strong diurnal patterns in the number of queries and the number of IP addresses submitting queries from this cluster over time, suggesting that these queries may be triggered by the events related to real users. For example, there could exist a front end receiving user requests, which triggers the back end to submit queries automatically. Each /24 IP prefix in this cluster has around 15 IP addresses, and the number of queries per host is well balanced, around 1000 queries per host.

This example shows different query contents and behavior patterns from the botnet cluster we study in Section 5.1. Without our tool, we would not be able to group the hosts or analyze their different intentions at a large scale.

5.3 Case Study III: Two Data Center Clusters Used For Malicious Purposes

Cluster$_1$ in D_1	
# of IPs	41
# of /24 prefixes	1
# of Queries	244054

Cluster$_2$ in D_2	
# of IPs	18
# of /24 prefixes	1
# of Queries	113994

Table 20. Case studies for two data center clusters

Although it is not surprising to observe botnets submitting queries with malicious intentions, the appearance of data center hosts submitting malicious queries is a relatively new trend. Using SBotScope we also find data center clusters submitting vulnerability-searching queries. Table 20 shows two example data center clusters (Cluster$_1$ and Cluster$_2$) from D_1 and D_2, both associated with the query pattern *"powered by"*, Cluster$_1$ has 41 IP addresses and Cluster$_2$ has 18 IP addresses, all from the same /24 IP prefix in AS21788 ("NOC"). All the IP addresses have been actively generating bot queries throughout the data collection period, with the query volume well balanced across hosts. Specifically, each host in these two clusters generated around 6000 queries, exhibiting strongly coordinated behaviors. Compared to botnets, data center IP addresses are often shared among multiple services, so they cannot be easily blacklisted.

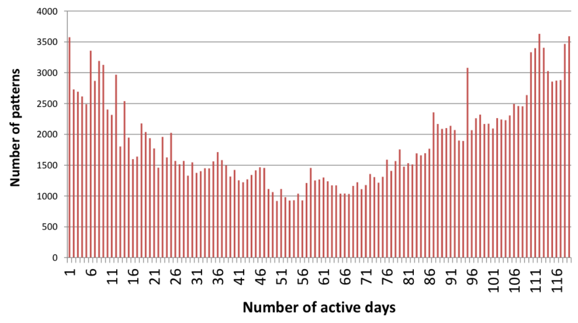

Figure 11. Query pattern lifetimes (from D_1)

5.4 Application: Detecting Additional Bot Queries

Since query patterns are more tightly related to the true intentions of bot queries, it is natural to consider using patterns for detecting bot queries that are missed by the existing detection approaches. However, two challenges exist for leveraging query patterns for detection. The first is whether query patterns are long-lasting enough so that we can apply the bot query patterns derived from historical data to newly observed ones. The second is whether bot query patterns are distinguishing enough to differentiate bot queries from those submitted by normal users.

To address the first challenge, we study the persistence of bot queries by checking the query patterns derived from D_1 against daily historical search logs in the past three months. Figure 11 shows that query pattern lifetime has a bimodal distribution. While many query patterns are relatively short-lived, there still exist a large number of query patterns that are persistent over time across almost the entire three months. Even for the short-lived query patterns, their lifetime is on the order of a few days, which still provide enough room for detection. In particular, we find many vulnerability-finding query patterns fall into the long-lived category [2]. This is encouraging as normal users usually do not search for specific web server vulnerabilities.

[2]We find a lot of query patterns for searching email account information fall into the short-lived pattern category

Newly detected bot queries	# of IPs	# of new IPs
12,436,658	3,873,909	3,361,532

Table 21. Newly detected bot queries in the same period of dataset D_2.

To address the second challenge, we eliminate query patterns that may introduce false positives. To be conservative, we use the following two rules for detection:

- Bot-query patterns should not match popular normal queries. Given that there could exist undetected bot queries, we match each pattern against historical normal user queries (e.g., three months ago). If a pattern matches normal user queries from a large number of hosts (≥ 1000 IP addresses), we do not select this pattern.

- A bot usually generates more than one query each time. To be conservative, we select only hosts who have at least m (we currently set $m = 100$, which is really conservative) bot queries (either already detected or newly matched) and mark their newly matched queries as additionally detected ones.

Applying these two rules on the normal query log collected in the same period as D_2, we additionally detected 12,436,658 bot queries that slip through the existing detection system. Table 21 shows that the newly detected bot queries have widely spread IP addresses, out of which, 86.8% (3,361,532) are addresses not observed in the already detected bot-query set.

To validate the newly detected bot queries, we take a similar approach by looking at the behavior similarity of these queries in their “Form”, “Referrer” and “UA” fields. To be comprehensive, we compare the similarity of these fields among newly detected queries sharing the same patterns, as well as comparing them against the clusters we already derived. Additionally we also compute the temporal correlations of their activities (i.e., activity burstiness). Using the combinations of these approaches, we can validate 94.13% as suspicious and likely undetected bot queries.

The purpose of this experiment is not to have a bulletproof new detection system, but to illustrate the value of exploring query patterns for detection. The hosts that submitted the newly detected bot queries were stealthier in their search behaviors, but their queries revealed their true intentions.

6 Discussion

Attackers may wish to evade any analysis that may lead to detection or defense. Since SBotScope directly focuses on query contents and the invariant patterns in terms of keyword combinations, typical behavior-based strategies such as randomizing UA, $Referer$ fields or mimicking legitimate query intervals will not impact our analysis.

Attackers may also wish to modify queries, increase or decrease the number of queries per host. However, doing so may lead to undesirable side effects. In particular, modifying queries may not retrieve the same quality search results. Increasing the number of queries per host (e.g., by adding obfuscated or randomly picked queries) may lead to detection as the hosts submit queries more aggressively. Reducing the number of queries will decrease the query throughputs, making it less cost-effective for attackers.

The ability to automatically dissect large-scale bot queries is only a first step toward leveraging the value of bot queries for security. As bot queries provide rich information about the interests and focus of attackers, we should analyze them further to extract stronger signals. The SBotScope system provides the necessary basis for further study. For example, we could analyze vulnerability-searching queries in more detail to reveal the vulnerable software and version numbers, the set of vulnerable web sites, and even zero-day exploits. Finally, the information about the botnet sizes and activity patterns can enable better detection and defense mechanisms.

7 Conclusion

In this paper, we have presented SBotScope, an automated system to analyze large-scale bot queries from two important aspects: *query intentions* and *query origination*. SBotScope performs syntactical and semantical analysis of bot queries to identify query intentions. It further clusters hosts and classifies them into botnet clusters and data center clusters. Our large-scale study, based on month-long bot queries collected from the Bing search engine, has revealed a number of unique findings. First we find that a significant portion of bot queries are associated with malicious activities, where 33% of bot queries are used for vulnerability discovery and 11% are used for harvesting email addresses. Second, SBotScope identifies 81,520 botnets involved in submitting bot queries on a global scale. Third, SBotScope identifies 287 data center clusters, some of which are also performing malicious searches. The appearance of using data center hosts for malicious activities seems a newly emerged trend. Finally, we demonstrate the value of SBotScope using a few case studies and a concrete application. We believe that SBotScope is a useful tool for improving security in multiple aspects.

8 Acknowledgements

We thank Zijian Zheng and his team for providing us with the Bing search logs. We thank Qifa Ke and Yunchao Gong for their help on the spectral clustering algorithm. We are grateful to Seungwon Shin and Guofei Gu for their help in validating IP addresses using the Conficker data set. The development of the SBotScope system and the related analysis were conducted while Junjie was at Microsoft Research Silicon Valley. The analysis on the comparison with the Conficker botnets was partially supported by the National Science Foundation under grant no. 0831300, and the Office of Naval Research under grants no. N000140710907 and no. N000140911042.

References

[1] Amazon.Com Server Said To Have Been Used In Sony Attack. http://www.bloomberg.com/news/2011-05-13/sony-network-said-to-have-been-invaded-by-hackers-using-amazon-com-server.html.

[2] Exploit DB. `http://www.exploit-db.com/google-dorks`.

[3] NOBISTECH. http://www.nobistech.net/.

[4] SOFTLAYER. http://www.softlayer.com/.

[5] Whois. http://en.wikipedia.org/wiki/Whois.

[6] L. M. Aiello, D. Donato, U. Ozertem, and F. Menczer. Behavior-driven clustering of queries into topics. In *Proc. ACM CIKM*, 2011.

[7] B. Miller, P. Pearce, C. Grier, C. Kreibich, and V. Paxson. What's clicking what? Techniques and innovations of today's Clickbots. In *Proc. DIMVA*, 2011.

[8] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *Proc. ACM SIGKDD*, 2000.

[9] D. Donato, F. Bonchi, T. Chi, and Y. Maarek. Do you want to take notes?: Identifying research missions in Yahoo! search pad. In *Proc. WWW*, 2010.

[10] D. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[11] E. Cooke, F. Jahanian, and D. McPherson. The zombie roundup: Understanding, detecting, and disrupting botnets. In *Proc. SRUTI*, 2005.

[12] F. R. Bach and M. I. Jordan. Learning spectral clustering, with application to speech separation. *Journal of Machine Learning Research*, 7:1963–2001, 2006.

[13] F. Yu, Y. Xie, and Q. Ke. Sbotminer: Large scale search bot detection. In *Proc. ACM WSDM*, 2010.

[14] G. Buehrer, J. W. Stokes, and K. Chellapilla. A large scale study of automated of automated web search traffic. In *Proc. AIRWEB*, 2008.

[15] G. Buehrer, J. W. Stokes, and K. Chellapilla. Classification of automated web traffic. In *Chapter in Weaving Services and People on the World Wide Web*, 2009.

[16] H. Cao, D.H. Hu, D. Shen, D. Jiang, J.-T. Sun, E. Chen, and Q. Yang. Context-aware query classification. In *Proc. ACM SIGIR*, 2009.

[17] H. Kang, K. Wang, D. Soukal, F. Behr, and Z. Zheng. Large-scale bot detection for search engines. In *Proc. WWW*, 2010.

[18] X. He and P. Jhala. Regularized query classification using search click information. *Pattern Recognition*, 41(7):2283–2288–145, 2008.

[19] J. P. John, F. Yu, Y. Xie, A. Krishnamurthy, and M. Abadi. deSEO: Combating search-result poisoning. In *Proc. USENIX Security*, 2011.

[20] J. P. John, F. Yu, Y. Xie, M. Abadi, and A. Krishnamurthy. Searching the searchers with searchaudit. In *Proc. USENIX Security*, 2010.

[21] J. Wen, J. Nie, and H. Zhang. Query clustering using user logs. *ACM Transactions on Information Systems*, 20(1):59–81, 2002.

[22] J.P. John, F. Yu, Y. Xie, A. Krishnamurthy, and M. Abadi. Heat-seeking honeypots: Design and experience. In *Proc. WWW*, 2011.

[23] L. Lu, R. Perdisci, and W. Lee. Surf: Detecting and measuring search poisoning. In *Proc. ACM CCS*, 2011.

[24] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *J. Intell. Inf. Syst.*, 17(2-3):107–145, 2001.

[25] N. Daswani and M. Stoppelman. The anatomy of Clickbot.A. In *Proc. USENIX HotBots*, 2007.

[26] N. Provos, J. McClain, and K. Wang. Search worms. In *Proc. ACM WORM*, 2006.

[27] S. M. Beitzel, E. C. Jensen, D. D. Lewis, A. Chowdhury, and O. Frieder. Automatic classification of web queries using very large unlabeled query logs. *ACM Transactions on Information Systems*, 2, 2007.

[28] S. Shin, G. Gu, N. Reddy, and C. Lee. A large-scale empirical study of Conficker. *IEEE Transactions on Information Forensics and Security*, 2012.

[29] S.M. Beitzel, E.C. Jensen, Q. Frieder, D.D. Lewis, A. Chowdhury, and A. Kolcz. Improving automatic query classification via semi-supervised learning. In *Proc. IEEE ICDM*, 2005.

[30] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov. Spamming botnet: Signatures and characteristics. In *Proc. ACM SIGCOMM*, 2008.