Spatial Statistics as a Metric for Detecting Botnet C2 Servers

Etienne Stalmans[1,2], Barry Irwin[2]

[1] SensePost Assessment Services, Pretoria, South Africa
etienne@sensepost.com
[2] Security and Networks Research Group, Rhodes University, South Africa
b.irwin@ru.ac.za

Abstract. Botnets consist of thousands of hosts infected with malware. As these hosts are widely dispersed and usually not physically accessible to botnet owners, a means to communicate with these hosts is needed. Using Command and Control (C2) servers botnet owners are able to communicate with and send commands to the members of the botnet with minimal effort. As these C2 servers are used to control the botnet, they can be used to shutdown the entire botnet by either by taking over or blocking the C2 servers. In defense to this botnet owners have employed numerous shutdown avoidance techniques. One of these techniques, DNS Fast-Flux, relies on rapidly changing address records. The addresses returned by the Fast-Flux DNS servers consist of geographically widely distributed hosts. These Fast-Flux C2 servers tend to be dispersed through multiple countries and across timezones. This distributed nature of Fast-Flux botnets differs from legitimate domains, which tend to have geographically clustered server locations. This paper examines the use of spatial autocorrelation techniques based on the geographic distribution of domain servers to detect Fast-Flux domains. Two means of measuring spatial autocorrelation, Moran's I and Geary's C, are used to produce classifiers. These classifiers use multiple geographic co-ordinate systems to assign unique values to each C2 server and subsequently to produce efficient and accurate classifiers. It is shown how Fast-Flux domains can be detected reliably while only a small percentage of false positives are produced.

Key words: security,botnets,geographic dispersion,spatial autocorrelation,DNS

1 Introduction

The number of Internet connected devices was estimated at 8.7 billion in 2012 and it has been predicted that this number will rise to 15 billion by 2015 [1]. The sheer number of devices and their diversity provides a large attack surface for malware. This has led to the emergence of botnets as a serious threat in the modern Internet landscape. As the number of infections continually rises it becomes evident that current defensive strategies are ineffective. Traditionally malware defense has focused on host based protection, with each host running a version of anti-virus that needs to be kept up to date. This requirement has meant that the management of malware protection technologies has become infeasiable. Furthermore, attackers are constantly finding new methods of bypassing anti-virus products, subsequently with each new bypass, anti-virus needs to be updated on each host. It is thus proposed that malware detection should be moved to the network layer where possible malware infections can be detected based on the traffic sent and received by hosts on the network.

While traditional botnets used Internet Relay Chat (IRC) servers as the communication channel between C2 servers and infected hosts, modern botnets have moved to HTTP and more recently HTTPS as the communication channel [2]. The use of HTTP and HTTPS allows botnet traffic to be disguised as legitimate Internet traffic. Furthermore, HTTP and HTTPS allows botnet traffic to traverse restrictive firewall rules as HTTP and HTTPS traffic is usually allowed through. The majority of inter-device communication on the Internet relies on the Internet Protocol (IP), this system assigns an address to each host that is easy for computing devices to understand but harder for humans to remember. The Domain Name System (DNS) was introduced as a solution to this problem and maps IP addresses to user-friendly domain names. Almost all network communication relies on DNS for address translation, prior to attempting to establish a connection. This reliance on DNS makes an ideal mechanism for detecting suspicious network communication before an actual connection has been established.

Botnet creators or operators aim to create large networks of infected hosts. These networks need to be managed and the botnet operators need methods for communicating with all the hosts of the botnet. Command and Control (C2) servers have emerged as a common strategy for managing these networks. C2 servers are usually hosted on bulletproof service networks or hijacked hosts [3]. Botnet operators have no physical control of these servers and can thus not alter the geographic location of these hosts. To prevent the C2 servers from being shut down, and with them the botnet, botnet controllers rely on different techniques. One of these techniques is known as Fast-Flux. Fast-Flux relies on DNS, where domains using Fast-Flux return multiple and rapidly changing IP addresses for C2 servers with each DNS query response. These servers normally consist of compromised hosts that are under the control of the botnet owners. These C2 address records map back to computer systems located world-wide [4]. The distributed nature of these C2 servers differs from legitimate domain servers. Legitimate domain servers under the control of organisations are usually located in a central geographic locations or are clustered in distributed cases. Furthermore, legitimate domains such as Content Distribution Networks, that do use geographically distributed servers display a degree of geographic intelligence, where users are directed to servers that are geographically close to them [5]. This distinguishing factor allows for the differentiation between legitimate DNS domain entries and Fast-Flux domain entries.

The work presented in this paper aimed to outline new classifiers based on animal and plant dispersion statistics, which allow for the accurate differentiation between legitimate and Fast-Flux domains. The classifiers produced are both accurate and lightweight, with no additional network traffic being generated and no time delay between a domain being seen for the first time and its classification. While this work primarily focuses on the detection of botnets that use Fast-Flux to mask the location of C2 servers, however, it should be possible to expand on these techniques to aid in the detection of other forms of botnet control channels. Furthermore, the techiniques outlined in this report could be used in other areas of network security, such as Distributed Denial of Service (DDoS) detection.

2 Background

Previously proposed techniques for detecting malicious network activity through DNS monitoring were studied. Observations made by these previous works indicated that a botnet detection system based on the geographic location of C2 servers could be effective. The detection techniques implemented by [6] aimed to detect malicious Fast-Flux service networks through the passive analysis of recursive DNS traffic traces. This work identified features of Fast-Flux DNS query results that were common across botnet families [6]. Common features identified were a short time-to-live (TTL), multiple Address (A) records and multiple ASNs. It was shown that the IP addresses associated with a Fast-Flux domain name were often from dissociated networks and changed rapidly. These dissociated networks tended to be spread accross geographic regions [6]. The definitive work in Fast-Flux detection was done by [7], who's work identified the same key DNS features. The primary observation from their work was that Fast-Flux botnets could be detected using the number of distinct A records returned and the number of different ASNs associated with the domain. Furthermore, results showed that botnet creators attempted to mimic the structure of Content Distribution Networks (CDNs). This similarity to CDNs often masked botnet activity and hindered the automatic classification of domains [7]. It was, however, noted that the inherent distributed structure of botnets could be used as a distinguishing factor. The authors of [7] further noted that botmasters have limited control over the hardware and network location of individual nodes, where CDNs have full control over the locations of nodes. Furthermore, botmasters could not easily obfuscate these geographic features.

The distributed nature of botnet C2's has been researched and has been suggested as a means of detecting and classificating botnets [4,5,8]. The research conducted by [8] proposed a method for delay-free detection of Fast-Flux service networks. This solution relied on spatial distribution estimation and spatial service relationship evaluation. The timezones each botnet node was located in was used to distinguish between different geographic system spaces and were combined with information entropy to measure how uniformly nodes were distributed. The work noted that benign domains tend to be distributed in the same timezone, while Fast-Flux nodes are widely distributed across multiple timezones. The authors further noted that if all the hosts of a botnet were to be located in the same timezone, timezone based entropy would not be an effective measure for detecting if the hosts belonged to a benign or Fast-Flux domain. The work performed by [4] aimed to model the behavioural patterns of Fast-Flux botnets. Using DNS records, they showed Fast-Flux botnets exhibit common characteristics, namely that botnets form clusters based on botnet size, growth and operation. Furthermore, it was shown that the majority of Fast-Flux botnets operate in more than five countries at a time, averaging between 20 and 40 countries. The global IP usage patterns of Fast-Flux botnets were studied offering a global perspective, with 240 nodes on four continents monitoring DNS traffic [5]. The results from this work found that Fast-Flux botnets advertise IP addresses from multiple countries, irrespective of where the DNS query came from, where as CDNs advertise IPs in a geographically aware manner. This observation provides valuable insight into the operation of Fast-Flux botnets, and further helps determine how a classifier that is capable of differentiating between Fast-Flux botnets and CDNs may be constructed.

3 Spatial Autocorrelation

The aim of this research was to find a way of classifying domains based on the geographical dispersion of the servers related to that domain. Statistics focused on modeling the dispersion patterns of plants and animals is a well understood problem and thus techniques from this area of research were applied to our botnet detection system. Spatial autocorrelation has been used in the study of animal dispersion patterns and allows researchers to measure the spatial dependence of locations within a geographic area. This measure of dependence is based on the First Law of Geography which states, ‘Everything is related to everything else, but near things are more related than distant things’ [9]. Thus, spatial autocorrelation can be described as the process of correlating the values of a single variable, strictly according to the proximity of other values in the same geographic space.

In statistics autocorrelation refers to the process of finding the correlation between points of a random process at different points in time. Autocorrelation is achieved by cross-correlating a signal with itself, effectively removing noise and revealing any obscured patterns hidden in the signal. Correlation is used to measure the dependence or statistical relationship between any two points in a distribution. This correlation can refer to any characteristic that the points share such as geographic location, value or dependence on other points. While autocorrelation measures the dependence of points in one dimension, time, spatial autocorrelation was developed to measure the dependence of points in two-dimensional space. Spatial autocorrelation allows for the correlation of points in time and space, along with multi-directional points.

For this work two means of measuring spatial autocorrelation were used; namely Moran’s Index and Geary’s Coefficient. These are described in greater detail below.

3.1 Moran's Index

Moran's Index (MI) provides a test for spatial autocorrelation in a set of continuous data . The MI is a weighted correlation coefficient using the distance between dispersed points as weights. The MI is based on the observation that points spatially closer together are more likely to be similar than points far apart [10]. Moran's index is calculated using the formula:

$$I = \frac{N}{\sum_i \sum_j w_{ij}} \frac{\sum_i \sum_j w_{ij}(X_i - \bar{X})(X_j - \bar{X})}{\sum_i (X_i - \bar{X})^2}$$

Where:

- I is the Moran Index
- N is the number of locations returned by the DNS query
- $X_n$is the value of the n^{th} variable of interest (timezonevalue, UTM value, MGRS value)
- $\bar{X}$ is the average of all values of N
- w_{ij} is the weight (distance) between two spatial points i and j

Algorithm 1: Moran's Index

Moran's Index (MI) relies on the observation that points closer together in geographic space tend to have more similarities in their attributes than points far apart. The values for I were calculated separately for legitimate and Fast-Flux domains. Once these values had been calculated they were compared to see if there were any distinguishing values which could be used for differentiating between the domains.

3.2 Geary's Coefficient

Similarly to Moran's Index, the Geary's Coefficient (GC) is used to measure spatial autocorrelation. The value of GC lies in the range [0-2]. Values between 0.0 and 1.0 indicates positive spatial autocorrelation while values between 1.0 and 2.0 indicate negative spatial autocorrelation. A value of 1.0 for GC indicates no spatial autocorrelation. The GC tends to be more sensitive to localised correlation and has been shown to be a good indicator of differences in small neighbourhoods. The GC is influenced far more than the MI by skewed distribution of numbers of neighbours and by outliers. The Moran's Index and Geary's Coefficient tend to give similar results and thus may be used in a two factor classifier. These two means of measurement are negatively related. The GC is calculated using the formula:

$$C = \frac{(N-1)\sum_i \sum_j w_{ij}(X_i - X_j)^2}{2W\sum_i (X_i - \bar{X})^2}$$

Where:

- C is the Geary Coefficient
- N is the number of locations returned by the DNS query
- X_n is the value of the n^{th} variable of interest (timezone value, UTM value, MGRS value)
- $\bar{X}$ is the average of all values of N
- w_{ij} is the weight (distance) between two spatial points i and j
- W is the sum of all w_{ij}

Algorithm 2: Geary's Coefficient

3.3 Measuring Geographic Location

The spatial autocorrelation methods that have been proposed required a means of measuring the value associated with a point in geographic space. As DNS query response features such as A records and TTLs vary greatly, other means of assigning values that could be used in calculating the spatial autocorrelation. Three means of assigning value were examined, these were timezones, the Universal Transverse Mercator system (UTM) and the Military Grid Reference System (MGRS).

Table 1. Geographic Data for a Fast-Flux Domain (*cjjasjjikooppfkja.ru*)

IP Address	Latitude:Longitude	UTM	MGRS
79.108.149.71	38.25:-0.7	37M	30SYH0125936055
79.139.110.20	49.7833:22.7833	39Q	34UFA2837416063
31.45.148.102	38.0:-97.0	37Z	14SPH7560307702
88.132.63.164	47.0333:19.7833	38Q	34TDT0755809583
124.6.3.225	22.6333:120.35	34Z	51QTF2762705352
89.229.214.126	53.7333:18.9167	39Q	34UCE6257855864

The timezone in which a server is located allows for a value that is easy to calculate and quantify. Using Greenwich Meridian Time (GMT) as the base value of zero, each timezone was assigned a value. GMT+1 was assigned the value of 100, GMT+2 the value 200 and so forth.

The UTM system allows the earth to be divided into sixty distinct zones, each zone representing a six-degree band of longitude. For UTM to be used in the calculation the UTM value needed to be converted to a fully numeric value. This was achieved by multiplying the numeric grid designator with the ordinal value of the alphabetic grid designator.

The MGRS is based on the UTM grid system and the similar Universal Polar Stereographic grid system. MGRS allows coordinates to be described as a grid point, down to one square meter. A MGRS grid point is identified by a grid zone designation, followed by a 100,000-meter square identification. Conversion of this grid value to a numerical value was achieved by multiplying the value V1 with the value V2. V1 was calculated using the first numeric grid descriptor multiplied with the ordinal values of the alphabetic grid descriptors. V2 was the integer value of the 100,000-meter square identification.

4 Dataset

The datasets used were divided into malicious (Fast-Flux) data and legitimate (safe) data. The legitimate domain data was obtained from the Alexa Top 1000 Global Sites list [11]. The Alexa dataset was cross-correlated with the Google Doubleclick Ad Planner Top-1000 Most Visited Sites list [12]. The malicious data was taken from multiple sources, including the Fast-Flux trackers for the ZeuS [13], obtained before the Microsoft take-down of ZeuS domains in March 2012 [14]. Furthermore, domains used by the Citidale, SpyEye [15] and other Fast-Flux botnets [16] were also used. During a brief resurgence in March 2012 the Hlux2/Kelihos botnet made use of Fast-Flux, data regarding these domains was obtained from a large European ISP. The final dataset used was divided into 1047 Fast-Flux domains and 1500 legitimate domains. Figure 1 shows the geographic distribution of botnet C2 Server IP addresses as observed in the malicious dataset.

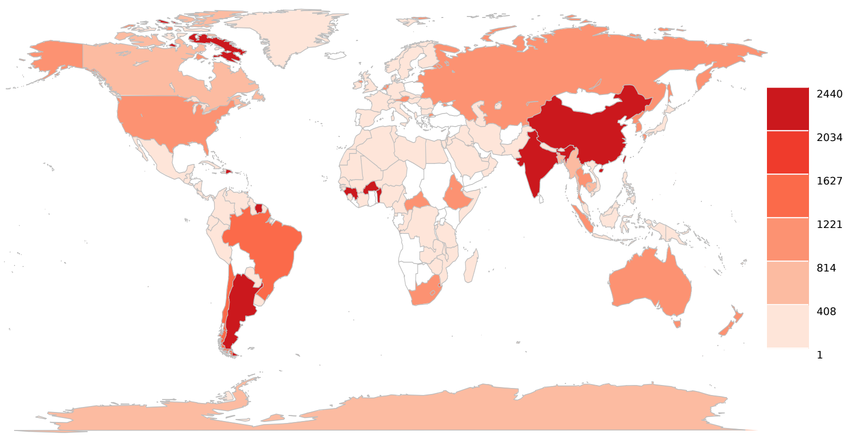

Fig. 1. Geographic Distribution of Botnet C2 Server IP Addresses

5 Results

Two means of measuring spatial autocorrelation were examined as classifiers, while three different measures of spatial value were used for each of these classifiers. Distance between geographic points were measured using the Haversine formula which measures the distance between two points on a curved surface.

5.1 Moran's I

The results for the MI classifier are shown in Table 2, where it can be seen that the classifier produced high TPRs using all three geographic position measures, while maintaining low FPRs. The overall accuracy for the classifiers was high with the lowest observed accuracy rate of 95%.

Table 2. Moran's Index Classifier Performance

	Accuracy (%)	TPR (%)	FPR (%)
Timezones	97	97	3
UTM	95	99	6
MGRS	95	99	6

Using timezones as the numerical value resulted in index values for legitimate domains that were generally zero, with 97% of observed legitimate domains having an index value of zero. The opposite held true for Fast-Flux domains, with only 3% of observed domains having an index value of zero. Using this as a classification boundary, domains were labeled as Fast-Flux if the returned index value was not equal to zero. When UTM was used as the value for C2 servers, it was observed that the index value for legitimate domains was zero, or very near zero, while Fast-Flux domains tended to have an index value of one or greater. Based on a classifier value of zero where any index value greater than zero indicated a Fast-Flux domain, a true positive rate of 99.03% was achieved. When using MGRS interesting results were obtained as the index value for Fast-Flux domains was distributed in clusters at -0.5 and zero. While the index value for legitimate domains was mostly zero, a smaller cluster formed around an index value of -1. Basing the classifier on the same logic as was used for the timezone and UTM classifier, with an index value of zero indicating a legitimate domain, a true positive rate of 99.35% was achieved. While a lower false positive rate of 6.02% was achieved. By modifying the classifier to classifying any value of -1 or 0 as legitimate led to an improved false positive rate of 1.24%, increasing the classifiers accuracy to 98.89%.

5.2 Geary's C

Geary's C (GC) is used for spatial autocorrelation, but is more sensitive to localisation than Moran's I. The use of GC should allow for classification of

Table 3. Geary's Coefficient Classifier Performance

	Accuracy (%)	TPR (%)	FPR (%)
Timezones	95	92	3
UTM	96	98	5
MGRS	95	99	6

spatial clusters in instances where Moran's I might not be as accurate. This might occur when the geographic location of servers are close together. The value produced by Geary's formula based on the timezone C2 servers were located in resulted in fewer Fast-Flux domains being correctly classified when compared with the MI classifier. The GC value was returned as zero for 96.58% of all legitimate domains, while returning a value greater than zero for 92.3% of Fast-Flux domains. Using this as the classification criteria, a classifier accuracy of 95.43% was achieved with a true positive rate of 91.83% and a false positive rate of 3.42%. The GC value for legitimate domains was clustered around zero as with the MI classifier, when UTM was used. The value for Fast-Flux domains displayed two clusters, one at 0.5 and a second around 1. This led to the use of the same classifying criteria as before, with a value of zero indicating a legitimate domain while a value greater than zero indicated a Fast-Flux domain. This resulted in a classifier with a 98.37% true positive rate and a low false positive rate of 4.56%. The overall accuracy for this classifier was a high 96.14%. The final classifier used MGRS as the value system and saw a similar trend to the MI classifier. The classifier decision boundary was set with a value of zero indicating a legitimate domain and any value above zero indicating a Fast-Flux domain. TA true positive rate of 99.64% was observed. The false positive rate of the classifier remained low at 6.02% and resulted in a classifier accuracy of 95.35%.

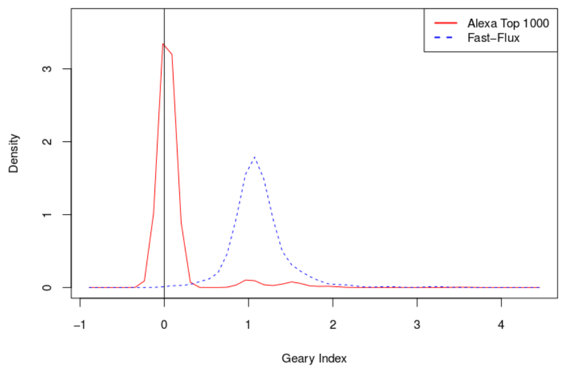

Fig. 2. Kernel Density Distribution for Geary's Coefficient.

5.3 Performance

The accuracy of the classifiers has been described in the preceding sections, however another critical aspect of evaluating the classifiers was the performance of the classifiers in terms of speed and resource consumption. These results are particularly pertinent to establishing the feasibility of using the proposed classifiers outside of the research environment and the potential for future implementation on live networks.

Testing was performed on an Intel Core i52410M 2.30GHz Ubuntu 12.04 x64 desktop PC, with 8 GB of DDR3 1600MHZ RAM. The classification system was coded in Python 2.7 and used a singlethreaded execution model. The live traffic dump file used, was read into memory in 20,000 packet batches and then passed to the classifiers. This was done to try avoid delays introduced when reading from disk, and also to minimise the effect of the inefficiency of reading .pcap files.

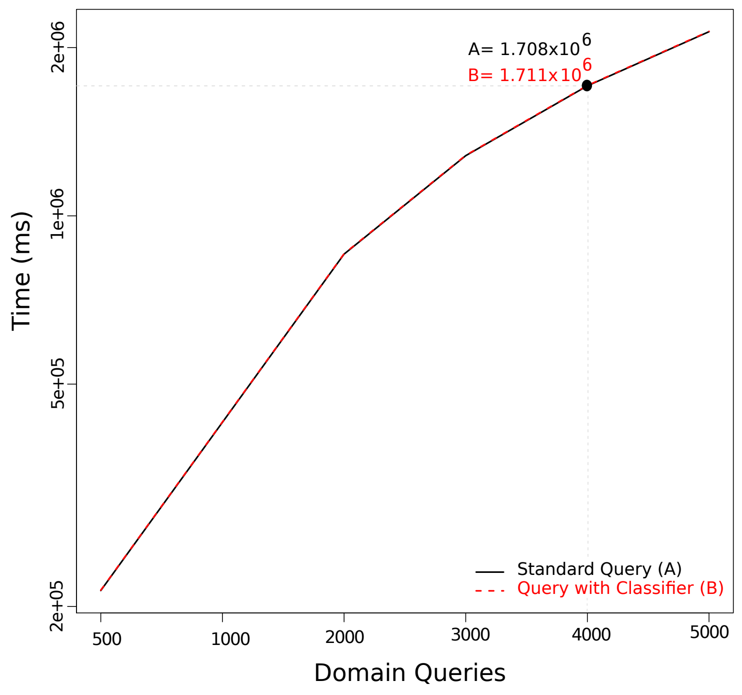

Fig. 3. Projected Performance Impact of DNS Query Response Classification

The data used for testing consisted of raw network traffic dumps obtained from the local schools in the region. The dataset consists of raw DNS query responses and represented the browsing habits of a typical small institute. It was felt that this would give a good representation of a real world scenario where the classifiers are used to process a network capture post-incident. Furthermore, it was predicted that the use of raw DNS query response captures would give a good indication of the expected processing overhead for analysing a DNS query response in realtime. The time it took to process 20,000 packets was measured where no classifiers where applied and where all the classifiers were used in sequence. Each test being repeated 600 times and the average processing time across these 600 iterations was used as an indicator of the classifier's overall performance. When all the classifiers were applied, processing 20,000 DNS query responses took 13.002 seconds and resulted in an average processing time of 6.5×10^{-1} ms for each individual DNS query response.

Finally the average time for an end-to-end DNS query was calculated by querying the top 500 domains in the Alexa dataset and it was determined that a DNS query took approximately 427.132ms to be completed. The DNS queries were repeated 4000 times to factor in DNS caching and the expected performance for classifying DNS query responses, compared to the calculated performance for standard DNS queries without classification are shown in Figure 3. It was noted that there was an indecernable increase in DNS query completion time, as can be seen in Figure 3, where 4000 standard DNS queries took 1.708×10^6 ms to complete and 4000 queries with the classifiers took 1.711×10^6 ms. This equated to an approximately increase of 0.152% in the time taken for a single DNS query to be completed.

6 Conclusion

Fast-Flux domains use multiple different servers located around the world to provide robust botnet control systems that are more resilient to shutdown attempts. This distributed nature provides a means for distinguishing between Fast-Flux domains and standard domains. This paper examined techniques for detecting Fast-Flux Command and Control (C2) domains. The techniques examined were able to accurately identify Fast-Flux domains based solely on the geographic locations of the C2 servers. The use of spatial autocorrelation has led to the creation of accurate and lightweight classifiers. Analysis was performed on known Fast-Flux domains, with the most accurate classifier correctly identifying 99.35% of domains as Fast-Flux. It was found that classifiers based on timezones produced the lowest number of false positives, while the use of the MGRS produced the best TPR. The best performing classifier was the Moran's Index classifier using timezones as a spatial measure, resulting in a classifier accuracy of 97% with a TPR of 97% and a FPR of 3%. The worst performing classifier was the Geary's Coefficient classifier using timezones, with an accuracy of 95% and a TPR of 92%, with a 3% FPR. Overall it was found that the classifiers are highly accurate and it was possible to detect up to 99% of all Fast-Flux domains with a high degree of confidence. There was a minimal trade-off between the TPR and FPR with the worst FPR observed being 6% but this was traded-off against a

TPR of 99%. Performance analysis was conducted to determine the feasibility of deploying the proposed classifiers in a real-world environment. It was shown that the proposed classifiers had a minor performance drawback in terms of time taken to process each domain. This performance impact was particularly negligible when compared to the time taken to perform a standard DNS query, with the classifiers adding only 0.152% to the execution time. With optimisation and the use of FPGA's the performance impact of the proposed classifiers could be further reduced.

The proposed solution provides an accurate means for improving network egress filtering, furthermore providing an effective additional layer of network defense usable in conjunction with existing defense systems. Future work will expand on the use of geographic location analysis, combining geographic location with other features identified as unique to Fast-Flux domains. Finally, it was shown that the predicted performance impact of the proposed classifiers would be minimal.

References

1. Forbes: How many things are currently connected to the "internet of things" (iot)? `http://www.forbes.com/sites/quora/2013/01/07/how-many-things-are-currently-connected-to-the-internet-of-things-iot/` (2013)
2. Morales, J.A., Al-Bataineh, A., Sandhu, R.: Analyzing DNS activities of bot processes. In: 2009 4th International Conference on Malicious and Unwanted Software (MALWARE), IEEE (2009) 98–103
3. Organisation, S.: Botnet statistics. `http://www.shadowserver.org/wiki/pmwiki.php/Stats/BotnetCharts` (2012)
4. Caglayan, A., Toothaker, M., Drapaeau, D., Burke, D., Eaton, G.: Behavioral patterns of fast flux service networks. In: Proceedings of the 2010 43rd Hawaii International Conference on System Sciences. HICSS '10, Washington, DC, USA, IEEE Computer Society (2010) 1–9
5. Hu, X., Knysz, M., Shin, K.G.: Measurement and analysis of global ip-usage patterns of fast-flux botnets. In: INFOCOM, IEEE (2011) 2633–2641
6. Perdisci, R., Corona, I., Dagon, D., Lee, W.: Detecting malicious flux service networks through passive analysis of recursive DNS traces. 2009 Annual Computer Security Applications Conference (2009) 311–320
7. Holz, T., Gorecki, C., Rieck, K., Freiling, F.C.: Measuring and detecting fast-flux service networks. In: MALWARE 2008. 3rd International Conference on Malicious and Unwanted Software, 2008. (2008) 24 – 31
8. Huang, S.Y., Mao, C.H., Lee, H.M.: Fast-flux service network detection based on spatial snapshot mechanism for delay-free detection. In Feng, D., Basin, D.A., Liu, P., eds.: ASIACCS, ACM (2010) 101–111
9. Tobler, W.: A computer movie simulating urban growth in the detroit region. Economic Geography **46**(2) (1970) 234–240
10. Lea, T., Giffith, D.: Opposites don't attract in spatial autocorrelation. GEO-World (14) (2001) 42–44
11. Alexa: Alexa top sites. `http://www.alexa.com/topsites` (2012)

12. Google: Top 1000 sites - doubleclick ad planner. http://www.google.com/adplanner/static/top1000/ (2012)
13. abuse.ch: Zeus monitor. https://zeustracker.abuse.ch/monitor.php?filter=level5 (2012)
14. Microsoft Corporation: Microsoft and financial services industry leaders target cybercriminal operations from zeus botnets. https://blogs.technet.com/b/microsoft_blog/archive/2012/03/25/microsoft-and-financial-services-industry-leaders-target-cybercriminal-operations-from-zeus-botne
aspx (2012)
15. abuse.ch: Spyeye monitor. https://spyeyetracker.abuse.ch/monitor.php?filter=level5 (2012)
16. ArborNetworks: Atlas global fast flux report. https://atlas.arbor.net/summary/fastflux (2012)