DNSSM: A Large Scale Passive DNS Security Monitoring Framework

Samuel Marchal[1], Jérôme François[1], Cynthia Wagner[1], Radu State[1],
Alexandre Dulaunoy[2], Thomas Engel[1], Olivier Festor[3]

[1] SnT - University of Luxembourg, Luxembourg – firstname.lastname@uni.lu
[2] Computer Incident Response Center Luxembourg, Luxembourg – alexandre.dulaunoy@circl.lu
[3] INRIA Nancy Grand Est, France – olivier.festor@inria.fr

***Abstract*—We present a monitoring approach and the supporting software architecture for passive DNS traffic. Monitoring DNS traffic can reveal essential network and system level activity profiles. Worm infected and botnet participating hosts can be identified and malicious backdoor communications can be detected. Any passive DNS monitoring solution needs to address several challenges that range from architectural approaches for dealing with large volumes of data up to specific Data Mining approaches for this purpose. We describe a framework that leverages state of the art distributed processing facilities with clustering techniques in order to detect anomalies in both online and offline DNS traffic. This framework entitled DNSSM is implemented and operational on several networks. We validate the framework against two large trace sets [1].**

I. Introduction

The **D**omain **N**ame **S**ervice (DNS) [1], [2] is one key component for the correct operation of the Internet. The most used functionality of DNS consists in binding a given URL (`www.example.com`) into its associated IP address (`123.4.56.78`). This step is called DNS resolution. An example, how the DNS resolution process works, is illustrated in Figure 1. Several threats specific to the DNS exist. These range from malicious domains hosting phishing sites and malware, covert channel communications over DNS to cache poisoning and client side attacks. We look in this paper at an approach for monitoring DNS traffic in order to mitigate some of the previously mentioned threats. More specifically, the main contributions presented in this paper are:

1) the design and implementation of a passive DNS monitoring architecture that can be used to track malicious domains,
2) the design of an automated clustering method that captures relevant groups of functional different domains,
3) some insights on content distribution networks from a local viewpoint.

The paper is structured as follows: Section II starts with a short introduction to security abuses in DNS. Next, section III describes the design and the architecture of the DNSSM framework. A detailed overview on the Data Mining specific part is presented in section III-B. The experimental results are provided in section IV. Related work is discussed in section V and section VI follows up with conclusions and discussions on future works.

[1]this work was partially made as part of an internship at INRIA Nancy Grand Est

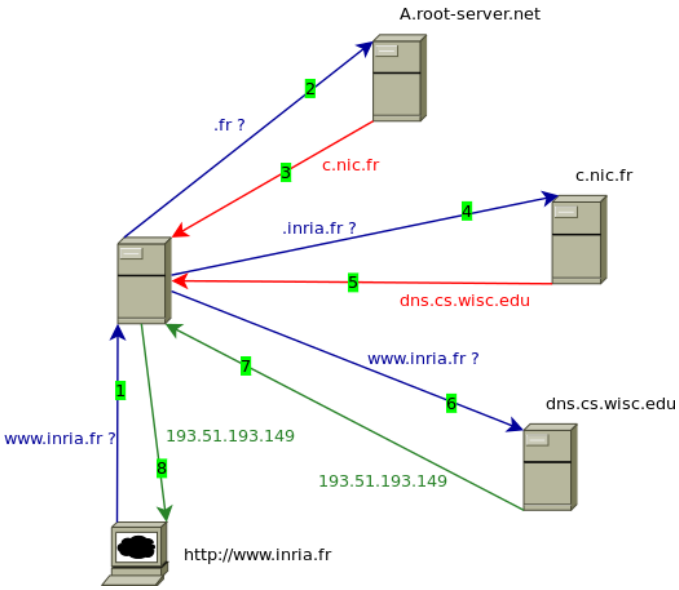


Fig. 1: DNS Resolution

II. Monitoring DNS Traffic

The Domain Name Service is a hierarchical distributed naming service in the Internet, where one or several servers are responsible for a given name space, called zone. For detailed information about DNS, the reader is referred to the references [1], [2]. The paper focuses on two major security abuses found in DNS. The first abuse consists in tunneling IP traffic over DNS traffic. Freely available tools like `dnstunnel` [3], implement this abuse and are widely used for bypassing WIFI Access Points authorization mechanisms. However, this tool can be much more nefarious, when it is used as backdoor from within an enterprise network. A second type of abuse, also called fast-flux (illustrated in Figure 2), consists in associating one fully qualified domain name with

several IP addresses. This technique is used for example for hosting phishing sites or command and control (C&C) centers of botnets [4], [5]. Rapidly changing DNS records and short-living TTL values in DNS replies ensure that, even if one of the IP addresses is removed from the network, the remaining ones guarantee the high resiliency of the infrastructure. The underlying idea of fast-flux is very simple. A combination of round-robin based registration and caching manipulation avoids any IP based access controls (firewall) or IP address oriented defensive measures. At a first glance, a simple fast-flux detection method could consist in monitoring DNS replies for large sets of different replies associated to one single domain name. However, this method will not work when faced with large server farms or content distribution networks (CDN) like `Akamai` [6]. Typical CDNs achieve the same redundancy as fast flux overlay networks, using similar techniques. The only subtle difference consists in the lifetime of a given domain name. Malicious domains have still short spanned life times. Since simple fast-flux networks can be taken down by removing the name server for the malicious domain, an improved architecture called double-flux emerged. This kind of architecture is illustrated in Figure 3 and avoids a single point of vulnerability by enabling dynamic updates of the name server entry list for a zone. Thus, multiple IP addresses, controlled by an attacker, can take the responsibility for the authoritative name server and thus, make the take-down of such a botnet much harder.

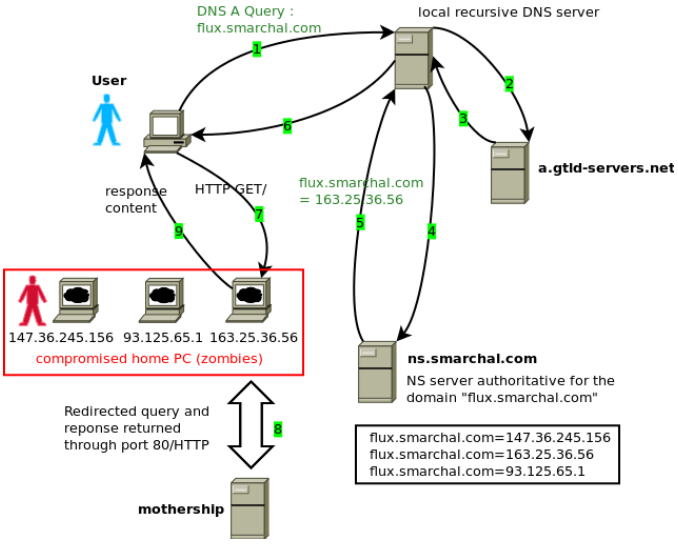


Fig. 2: Single Flux Networks

III. Gathering technique of DNSSM

In this section, the architecture and the design of the passive monitoring solution DNSSM are described. This architecture has to comply with several major requirements. From an operational point of view, it has to be able to retrieve both online and offline DNS packet captures. Thus, it can be used as an online monitoring tool, but can also be applied as an offline incident handling tool. For instance, in case

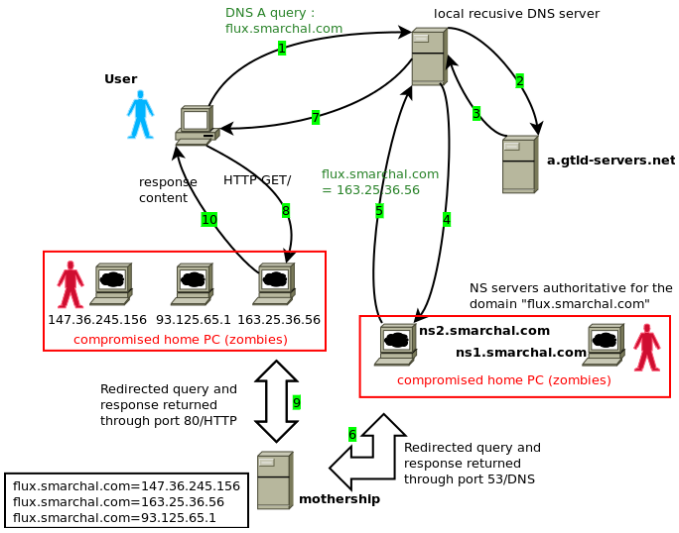


Fig. 3: Double Flux Networks

of analyzing a compromised network, the tool should be able to retrospectively analyze DNS traffic, detect and report suspicious activities.

A. The Architecture

The core DNSSM architecture (illustrated in Figure 4) comprises three main components and is based on the architecture proposed by Florian Weimer in [7]. The first component is a passive DNS sensor that is a simple packet capturer filtering DNS related traffic. This sensor should be placed between the recursive DNS server and the upstream DNS servers and its purpose is to listen for DNS replies, filter data and feed the retrieved information into the centralized storage which is here a relationnal databse system (MySQL). This sounds simple in theory, but in practice this tends to be more complex. Many DNS replies are not well structured and many reply messages have been observed to be erroneous. For instance, we have observed large quantities of `A` Record types that were returned to `127.0.0.1`. Therefore, a tedious case by case analysis had to be performed. Another unexpected issue consists in letter capitalization. A question that arises here is, if names of domains should be normalized to small capitals or if there may be large capitals too. This questions sounds obvious, but we have observed that some big 'actors' in the Internet (i.e. *Google*) play with variations within the same name, with both, small caps and large caps. For instance for *google.com*, respective PTR records can have both kinds of caps, some examples are, *GoOgle.com*, *gOOgle.com*, *gOoGle.com*, etc. The assumptions regarding this behavior are that Google somehow uses this trick to limit the impact of cache poisoning (in case of badly implemented DNS cache server), to infer the origin of the answers (geolocalization of a datacenter e.g.) or even to encode some data back from the original query of the user. Data that is stored, can be analyzed by both, a human operator using a Web based interface or automatically be mined by a Data Mining application. In order to make the

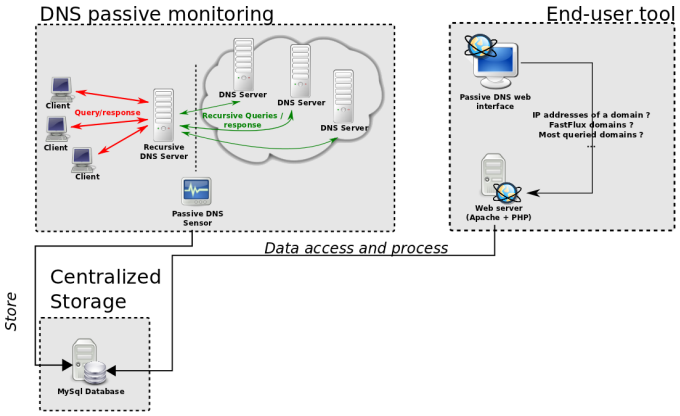


Fig. 4: DNSSM Architecture

tool more flexible, we have designed both variants, where we have applied the `R`-programming language [8] and the popular Data Mining tool `Weka` [9].

B. Data Mining in DNS Space

For the Data Mining evaluation of our approach, we had to define different analysis parameters in order to model DNS information. Therefore, we define ten different relative features. We filter and retain only the different domain names that have been requested during an observation period and for which at least one reply with the RCode *NOERROR* was received. These features and their respective motivations can be described as follows:

- The number of IPv4 addresses associated with the same domain names: we search for `A` and `CNAME` records from gathered Resource Records (`RR`) and count all IP addresses associated to the domain name. This aims at detecting flux networks or content distribution networks, which have high counts.
- An entropy-based index of IP address scattering S_{ip1} : the aim of this index is to show the scattering of IP addresses associated to the same domain name through different subnets on the internet. The aim is to locate all IP addresses for a domain name and to rewrite them in binary form. Then, for all 32-bit positions of the IP address, we calculate the Shannon entropy [10] that respects the two conditions : the bit is 0 or the bit is 1. To recall, a Shannon entropy for a variable x is defined as: $H(X) = -\sum_x p(x) log_2[p(x)]$ with $p(x)$ being the probability that X is in state x and $Plog_2P$ set to 0 if $p = 0$. After this action, we sum the 32 entropy values to obtain an index between 0 and 32, where 32 represents the maximum scattering.
- S_{ip2} represents the second IP address scattering index: The advantage of S_{ip2} is that it considers two things, which are not used in the first index calculation (S_{ip1}). Here, the differences in the positions among IP addresses are weighted, since it can intuitively be said that the first byte in an IP address has a higher relevancy and by this a higher weight than the rest of the IP address. Second, it also considers/counts the amount of different IP addresses in a data set. Regarding the entropy, if we have two addresses `0.0.0.0` and `255.255.255.255`, the index S_{ip1} will be at maximum value (i.e. 32), but if we have 20 different IP addresses for example, the index will be lower, even if we can observe that the scattering becomes more important. S_{ip2} attributes different weights to individual byte positions. For each of the 3 first positions of IPv4 addresses, we count the number of different bytes in the IP address pool we extracted for a domain name and then transform the obtained value into the base of 10.

$$S_{ip2}(\text{domain name}) = 100\times \text{ nbdifferentsbytes}_1 + 10\times \text{ nbdifferentsbytes}_2 + \text{nbdifferentsbytes}_3$$

- The mean `TTL` value for `A` domain records. Small `TTL` values are usually a useful indicator for flux-networks.
- The total count of requests made over the observation period for domain names. Variations in the distribution of this variable might indicate that an anomaly (e.g. phishing site) occurred.
- The day period during which we can observe requests for a domain. If there is only one request observed, this feature is set to 0. It has been observed that legitimate domains have longer time spans, while malicious domains exhibit daily/hourly lifetimes.
- The ratio for requests per time period. This is a combination of the two previous features, which provides the number of requests on a per hour base. This statistical feature captures the average usage pattern/frequency for a domain. It's a usefull parameter in order to detect for example domains involved in botnet as they are regularly requested over a time period.
- The sub-domain count for a given domain. This parameter, when it has a high value, can disclose algorithmically generated domain names for instance.
- The count of authoritative servers for a domain. For this feature, high values are a clear indicator that there is a double flux network.
- A binary indicator is set to 0, if a domain name is not blacklisted, otherwise this parameter is set to 1. This indicator is based on available data about regularly updated domain blacklists that can be freely downloaded from the Internet[2]. This parameter is only used for the needs of the web interface but not for classification.

IV. Experimental Results

A. Data Sets

In this section, the experimental outcomes for two different data sets collected in June 2011, are presented. We aimed at comparing very different data sets, different in the sense of geographical location, volume, duration and access networks.

[2]www.malwaredomainlist.com, www.malwaredomains.com, zeustracker.abuse.ch

Country	Luxembourg	France
Duration	1 hour	1 hour
# DNS replies	10 M	70095
Size of Datat	270MB	22.8MB

TABLE I: Passive DNS capture statistics

The first data set is originated from the INRIA Nancy Research Labs, which represents a medium sized campus network. The second data set originates from a regional Internet Service Provider located in Luxembourg. The quantitative differences between the two data sets are regrouped in Table I.

Table II summarizes some different viewpoints from the two datasets. We have looked at how a set of hosts and domains vary in the two datasets. For the domains, we have selected some poplar and well represented domains like: `Akamai`, `Facebook`, `Apple`, and respectively `Google`. From this table, we infer that both datasets capture a similar number of different subdomains for three domains `Facebook`, `Apple`, and `Google`, but a significant difference exists for the domain `Akamai`. This is natural since `Akamai` is a major CDN that hosts services which are much more represented in the larger dataset. Some recent estimates [11] of the `Akamai` network range it at about 50000 hosts world wide. Although a passive DNS analysis will only reveal geographically closed `Akamai` hosts, this data is relevant to estimate the local load balancing and service availability of the `Akamai` CDN.

Measure	data set 1	data set 2
# IP for mx.twitter.com	20	0
# IP for csi.l.google.com	74	404
# IP for star.facebook.com	8	39
# IP for x.apple.com.akadns.net	24	24
# subdomains for Akamai	1137	135
# subdomains Google	306	444
# subdomains for Facebook	174	66
# subdomains for Apple	134	156

TABLE II: Domain measures

B. Data Mining Approach

The Data Mining approach, is based on a clustering task. For this, it has been referred to the open-source Machine Learning tool `Weka`. This tool is known for its large library of supervised and unsupervised Machine Learning algorithms for real scenarios.

For the Data Mining experiments performed on the data sets, groups of domain names that share common behaviors should be identified. Therefore, the values obtained by the set of different features have been applied to the k-means algorithm [12]. The k-means algorithm is a classical clustering algorithm that is commonly used in Data Mining. For the detailed k-means algorithm we refer the reader to [12]. The aim of k-means is to divide instances into k clusters. More formally this means, given a set of instances (x_1,..., x_n), with each instance being a d-dimensional vector, k-means tries to optimally divide n instances into k clusters S={S_1,...,S_k}, where k has to be

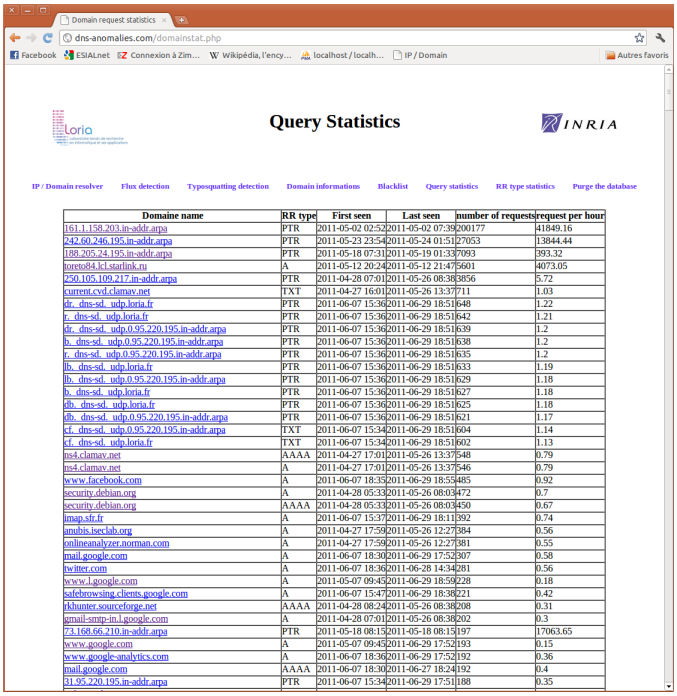


Fig. 5: DNSSM Online User Interface

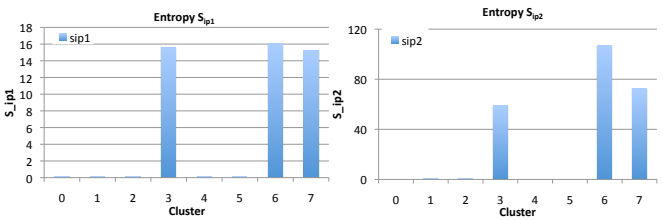
set in advance and (k $\leq$ n) and the intra-cluster sum of squares has to be minimized. This can be expressed as follows,

$$argmin_s \sum_{i=1}^{k} \sum_{x_\in S_i} ||x_j - \mu_i||^2$$

where μ_i represents the mean for the instances in S_i. For the analysis, the value for k has been set to $k = 8$, as for this amount of clusters, the best results were obtained.

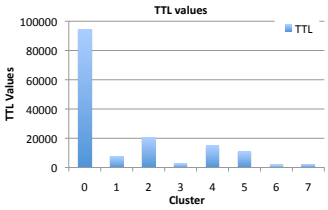
C. Results

In the experiments, we have analyzed the relations between the obtained clusters and the different features. Three clusters, cluster 3, 6, 7, are associated to high values in the S_{ip1} values (Figure 6a) and very similar low values for the TTLs (Figure 6c). The S_{ip1} values are two orders of magnitude higher than values for the remaining clusters. When looking at the cluster membership, we have observed that cluster 7 is grouping domains that perform user tracking services, like for instance *doubleclick.net*, *tradedoubler.com*, *quantcast.com*,.... Cluster 6 stands for very popular domains like *google.com, facebook.com, skype.com*, etc. These domains are operated over largely dispersed IP range spaces and thus exhibit a high dispersion. Cluster 3 is particular. While clusters 6 and 7 have both dispersion metrics in the high ranges (S_{ip1}, S_{ip2} in Figure 6a and 6b), cluster 3 groups domains characterized by high values in the S_{ip1} and lower (compared to clusters 6 and 7) values in the S_{ip2} metric. A manual analysis of this cluster showed that most of the domain names are related to CDNs as for example Akamai or Cloudfront. Typical domain names in this cluster have the following pattern, *fbcdn-video-a.akamaihd.net ou d19n4gh4cmsbnt.cloudfront.net*. This result is consistent with the operation of CDNs, where DNS replies

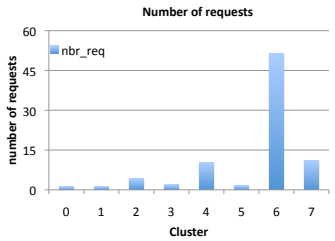


(a) Entropy S_{ip1}

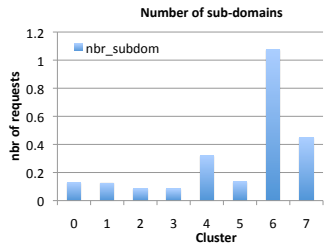
(b) Entropy S_{ip2}



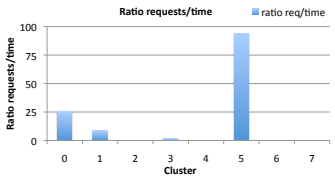
(c) TTL



(d) Number Requests



(e) Number Subdomains



(f) Ratio Requests per time

Fig. 6: Evaluation for some features

are based on geographical locations of the requesting client. In our case, we have observed high values in the number of `A` records per domain name, but most of the returned values were discriminated by the S_{ip2} metric.

Cluster 5 regroups domains with large numbers of requests per time interval (Figure 6f). These domains are very popular with very high ranks by the `alexa.com` ranking. Typical domain names in this cluster are, *apple.com, amazon.fr* or *adobe.com*.

Malicious domains (80) were also present in one dataset. Domains in this list were clustered in cluster 7. Examples of such domains are:

- 00007.ru, used for hosting malicious code
- 000.bbexe.cn, used for phishing site and rogue login script
- 01.finni.in, used for malicious hosting
- 010608.myftp.biz, used to host infected pdf files that exploit an Adobe Acrobat Reader vulnerability

We have looked at the statistical profile of DNS record types. We follow a sliding window approach, in which basic frequency counting is done for each window. Successive windows can then be compared using a statistical test to check for matching distribution. For instance, tunneling over DNS was easily detected because the proportion of TXT records increased significantly. This is due to the operation of dnstunnel, where the payload is encapsulated in TXT records. Table III shows a typical distribution from out ISP generated dataset. Worm and botnet spam activity can be identified by the checking the proportion of MX entries. Significant increase in this statistics is a clear sign of some mass mailing campaign.

RR Tpype	Proportion
% A	0.49
% NS	0.003
% AAAA	0.085
% MX	0.093
% CNAME	0
% SOA	0.0011
% PTR	0.2728
% TXT	0.0426
% SRV	0.0001

TABLE III: Distribution of record types

Looking into the specific data entries can reveal highly interested information. For instance, in our dataset, the SRV entries were mainly associated to VoIP gateways. The proportion of AAAA records can also give a hint on the adoption of IPv6.

D. User Interface

Figure 5 shows a general appearance of the User Interface of DNSSM. In this GUI, the analyst can compare different domains by exploring them with help of the features (see section III-B). A complete kit for own installation can be retrieved [3] under a GPLv2 license.

V. Related Work

Historically, the pioneering work of Florian Weimer [7] was the first description of a passive DNS monitoring toolkit that can be supported by additional data extraction tools. Some of the proposed extraction tools used linguistic techniques that use language processing models [13] for detecting malicious domains. This approach might be useful in the detection of automatically generated domain names without referring to human driven naming schemes. In practice, malicious domains exhibit naming schemes that are indistinguishable from normal domains. In [13], this work was extended by an additional module to track IP addresses, which are associated to given domains. The detection of tunneling over DNS was described in [14]. One of the first articles to address the relevance of malicious DNS traffic is [15], where the impact of corrupted DNS paths in the current Internet are illustrated.

Dealing with large quantities of DNS data has been addressed in [16] and [17], where tree-like structures are used to store compressed DNS replies and context-driven clustering to isolate malicious domains.

A somehow alternative idea is discussed in [18], where a reputation framework is proposed to cluster name servers, based on an inferred reputation. The placement of a passive DNS probe has been subject of research for its own in [19], [7], [18], [20], [21], [22] and [23], in order to determine

[3] https://gforge.inria.fr/docman/view.php/3526/7602/kit_dns_anomalies.tar.gz

only DNS traffic that has been exchanged between a recursive server and the respective authoritative domain servers. Target applications addressed by these works are the detection of phishing domains. Capturing the original DNS requests issued by end users is discussed in [17], [24], [15] and [25]. While timing related information can be integrated into the DNS monitoring approaches, privacy concerns related to affected end-users are hindering a large scale deployment.

Machine learning techniques for mining DNS data are proposed in [26], [20], [21]. The main difference to our work is that our approach leverages unsupervised clustering techniques and neither relies on labeled or annotated data sets, nor does it require external blacklists.

VI. Conclusion and Future Work

As a core service of the internet, DNS carries a huge amount of information that is extremely rich to the security monitoring activity. To exploit this information in an efficient and automated way, we have presented in this paper a passive DNS monitoring solution that leverages a relevant Data Mining algorithm. In this work, we share our observations of DNS activity and operational results from DNS monitoring by presenting a novel well designed and implemented prototype, called DNSSM. DNSSM is an open-source tool that can be freely downloaded. Though the framework, we have analyzed two different datasets of passively collected DNS traces by applying an automated clustering algorithm, where the different clusters represent different types of DNS traffic activities. The efficiency of this analysis has been shown in the paper. The practical outcomes are multiple. Firstly, DNSSM can be used to dig in the namespace and expose some clear patterns of DNS deployment. Features related to the TTL, number of A records and lifetime of a given domain can assist in investigating cyber security frauds. Secondly, DNSSM can be used to debug a DNS deployment. It can provide a local view on how records from a domain are seen by a community of users. One important side functionality is its capability to map content distribution networks. Passive monitoring of large ISP domain, will infer a large part of current CDN deployment and sizes. Finally, penetration testing and security assessment operations can leverage data obtained from DNSSM in order to map the target network.

Our future work will consist in extending this analysis approach by using more data sets, where a ground truth of several botnet infected hosts will be set up. In addition to new traces, we also plan to enrich the approach with new features to characterize the traffic so that data can be analyzed more intuitively. We are also looking forward toward correlating passive DNS data with Netflow monitoring, albeit the setup of such an experiment is quite complex and needs to comply to existing privacy protecting legislation.

Acknowledgments

We want first to acknowledge CETREL for its financial support in these research. We want to address our special thanks to RESTENA Luxembourg for their support. This work pas partially supported by the french ANR VAMPIRE project and some experiments where performed within INRIA's High Security Lab.

References

[1] P. Mockapetris, "Rfc 1034: Domain names - concepts and facilities," 1987.
[2] ——, "Rfc 1035: Domain names - implementation and specification," 1987.
[3] K. Dan, "Ozymandns : Kaminsky dns tunnel," 2005.
[4] DAMBALA, "The command structure of the aurora botnet," March 2010.
[5] ——, "Botnet communication topologies," 2009.
[6] "Akamai cdn," http://www.akamai.com/.
[7] F. Weimer, "Passive dns replication," 2005.
[8] CRAN, "R-project," http://www.r-project.org.
[9] H. Mark, F. Eibe, H. Geoffrey, P. Bernhard, R. Peter, and W. Ian, "The weka data mining software: An update," 2009.
[10] C. E. Shannon, "A mathematical theory of communication," *Bell system technical journal*, vol. 27, 1948.
[11] C. Huang, A. Wang, J. Li, and K. W. Ross, "Measuring and evaluating large-scale cdns paper withdrawn at mirosoft's request," in *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*, ser. IMC '08. New York, NY, USA: ACM, 2008, pp. 15–29. [Online]. Available: http://dl.acm.org/citation.cfm?id=1452520.1455517
[12] J. A. Hartigan and M. A. Wong, "A k-means clustering algorithm," *Applied Statistics*, vol. 28, 1979.
[13] S. Yadav, A. K. Reddy, N. Reddy, and S. Ranjan, "Detecting algorithimically generated malicious domain names," 2011.
[14] K. Born and D. Gustafson, "Detecting dns tunnels using character frequency analysis," *CoRR*, vol. abs/1004.4358, 2010.
[15] D. Dagon, N. Provos, C. P. Lee, and W. Lee, "Corrupted dns resolution paths: The rise of a malicious resolution authority."
[16] A. Hunt, "Visualizing the hosting patterns of modern cybercriminal," September 2010.
[17] D. Plonka and P. Barford, "Context-aware clustering of dns query traffic." in *Internet Measurement Comference'08*, 2008, pp. 217–230.
[18] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, "Building a dynamic reputation system for dns," in *Proceedings of the 19th USENIX conference on Security*, ser. USENIX Security'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 18–18.
[19] B. Zdrnja, N. Brownlee, and D. Wessels, "Passive monitoring of dns anomalies," in *Proceedings of the 4th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, ser. DIMVA '07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 129–139.
[20] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, "Finding malicious domains using passive dns analysis," in *NDSS'11, 18th Annual Network & Distributed System Security Symposium, 6-9 February 2011, San Diego, California, USA*, February 2011.
[21] B. Zdrnja, "Security monitoring of dns traffic," May 2006.
[22] M. Antonakakis, D. Dagon, X. Luo, R. Perdisci, W. Lee, and J. Bellmor, "A centralized monitoring infrastructure for improving dns security," in *Proceedings of the 13th international conference on Recent advances in intrusion detection*, ser. RAID'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 18–37.
[23] J. M. Spring, "Large scale dns traffic analysis of malicious internet activity with a focus on evaluating the response time of blocking phishing site," Master's thesis, University of Pittsburgh, 2010.
[24] R. Perdisci, I. Corona, D. Dagon, and W. Lee, "Detecting malicious flux service networks through passive analysis of recursive dns traces," in *Proceedings of the 2009 Annual Computer Security Applications Conference*, ser. ACSAC '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 311–320.
[25] H. van der Heide and N. Barendregt, "Dns anomaly detection," 2011.
[26] *EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis*, Internet Society. Internet Society, Feb. 2011. [Online]. Available: http://www.iseclab.org/papers/bilge-ndss11.pdf