Identifying the True Origin of DNS Traffic Without Reference to Client Source Address

Joe Abley
Western University, London, Ontario, Canada
Afilias Canada, Toronto, Ontario, Canada
jabley@uwo.ca, jabley@afilias.info

***Abstract*—We demonstrate a classifier system that is able to identify the originating system respobsible for stateless Domain Name System (DNS) traffic received at an authoritative DNS server without reference to source address. The ability to determine whether particular DNS query traffic received at an authoritative server is legitimately sourced from a particular client system is useful in identifying some classes of malicious traffic in production DNS systems.**

I. Introduction

The Domain Name System (DNS) includes a wire protocol with which structured requests and responses are exchanged over a network. The DNS protocol was originally specified [1] [2] for use over both the Transmission Control Protocol (TCP) [3] and the User Datagram Protocol (UDP) [4] and the use of other transports have also been documented [5] [6] [7]. At present, however, UDP is the overwhelmingly dominant transport protocol in use; for example, according to statistics published by ICANN for queries received at the L root server, UDP accounts for 98% of all queries received[1].

Since UDP transport for DNS is stateless, consisting of single-datagram queries and responses with no setup or tear-down handshake, there are limited opportunities to verify the legitimacy of a source address. DNS servers are consequently frequently used as amplifiers in reflection attacks [8]. Although some such attacks are trivially identified, e.g. by Query Type (QTYPE), many are more difficult. By choosing query parameters that match legitimate, real-world use of the DNS, attackers can make it difficult for their traffic to be identified and blocked without causing collateral damage. This is especially true of amplification attacks against DNS resolvers.

The clients of authoritative DNS servers are most usually DNS resolvers. These client systems receive requests from end-user applications (or downstream resolvers). Different client resolver systems are observed to send different mixes of DNS traffic; for example, a resolver system that mainly serves end-users will send a different mixture of queries to authoritative servers than one which serves a specific application like Internet mail [9], which might reasonably be expected to have a much higher proportion of query traffic with `QTYPE=MX`.

[1]http://stats.dns.icann.org/plotcache/L-Root/transport_vs_qtype/2018-12-03T00:00-2018-12-03T23:59-all.html

Afilias Canada[2] operates authoritative DNS infrastructure for around 300 top-level domains, including several that attract high levels of query traffic such as INFO and ORG. This infrastructure is distributed globally using anycast service distribution [10], using commodity transit services, public peering and so-called Private Network Interconnects (PNIs). The real origin of queries received over a PNI can be known with high accuracy; the origin of queries received over the Internet, in contrast, cannot. We refer to the former as *trusted* paths, and the latter as *untrusted*. Trusted paths exist to Google Public DNS[3], a public DNS resolver system configured for use by a large number of end-users, and Facebook[4], whose resolver systems are mainly used by back-end systems that build previews for links shared between users of Facebook's social media platform. The traffic patterns of each are expected to be usefully different.

While real-time anomoly detection in DNS traffic remains an elusive problem, the ability to classify traffic apparently received by particular sources as being legitimate is useful in the forensic analysis of traffic spikes since it provides the opportunity to distinguish between illegitimate, unwanted traffic and traffic from clients that just happen to be busy, e.g. due to a burst in popularity in a particular web page, or changes in the Time To Live (TTL) parameters of high-use domain names. This paper describes a system that aims to provide such a classification.

A raw DNS dataset is collected in the form of individual (request, response) DNS messages received and sent from and to a single apparent source over a time period. We extract features from those messages and count them over a short time interval to build a vector of features that describes the traffic during that time. Each such vector, once normalised, represents a single observation. Sets of observations can be collected from adjacent sample intervals. Where the traffic arrived from a trusted source, the source system can be identified and included in the dataset which can subsequently be used for training. Traffic that definitively did not arrive from a trusted source can be used to classify traffic as "other". The resulting model can be used to classify query streams from unknown sources to

[2]https://afilias.info/
[3]https://dns.google.com
[4]https://www.facebook.com

classify the origin of the query traffic as "Facebook", "Google" or "Other".

This paper is organised as follows. Section I introduces the problem and provides some high-level background on the DNS. Section II provides a short introduction to the algorithms and accuracy measures that are used to build the model. Some other work on applying machine learning techniques to problems in the DNS are described in section III. Data collection and preprocessing, feature engineering and choice of learning and validation algorithms are discussed in section IV. Section V describes the evaluation of the resulting model; section VI provides a summary.

II. Background

Two multiclass classifiers are evaluated for this model in section II-A, below. The accuracy of each is assessed as described in section II-B.

A. Classifier Methods

1) Multiclass Support Vector Machine: The classifier used in this paper was constructed as a series of Support Vector Machines (SVM), each used as a binary classifier. SVM represents n-dimensional support vectors in an n-axis hyperspace and identifies a hyperplane boundary between observations known to be in different categories to facilitate classification of unlabelled test sets. Those boundaries can then be used to classify unlabelled observations.

Multiclass classification is achieved using $k(k-1)/2$ *one-against-one* binary classifiers combined with a max-wins voting scheme, as discussed in [11].

The SVM implementation used to construct this model exposes several hyperparameters that can be tuned, as well as a native grid search to assist identification of optimal parameters for a supplied validation dataset.

2) Random Forest: Random Forests (RF) [12] combine many decision trees at training time into an ensemble learning model. RF is an improvement over the use of individual decision trees since they are far less susceptible to over-fitting; in fact, Breiman asserts that RF in general does not over-fit, although it is not clear that the assertion is supported by widely-accepted analysis in the general case.

The RF implementation used to construct this model exposes several parameters that can be tuned, including the maximum tree depth, the maximum number of nodes and the bootstrap sample size.

B. Accuracy

Short introduction to the algorithms and the accuracy measures used. Generic only, and nothing about how the algorithms were used or the data. Half page, max one page.

III. Related Work

Machine learning techniques were applied to the problem of classifying so-called core domains as part of a threat assessment in a production system at Nominet[5] [13] [14]. This problem has some similarities to the problem described in this paper, and illustrates the use of continuous learning to upadate an already-trained model on arrival of new data.

The .NZ registry maintains a set of business intelligence datasets which are constructed in part by analysis of queries received at authoritative DNS servers. In order to improve the accuracy of those datasets, machine learning techniques were used to build models that could classify query sources as DNS resolvers or other systems (e.g. systems performing active monitoring of the DNS). The work included extensive feature analysis and incorporated substantial domain knowledge derived from earlier analysis. [15] [16].

A study in the application of different machine learning techniques was presented in [17] as part of an attempt to train a model to identify Internet traffic tunnelled over the DNS protocol.

The approach described in this paper differs from other approaches described above in that it acknowledges the problems inherent in grouping DNS transactions together without the ability to be certain that the apparent sources of DNS queries are legitimate.

IV. Methodology

A. Overview

Complete sets of query data received over both trusted and untrusted paths were collected at a representative number of Afilias anycast sites, in the form of packet captures in PCAP format.[6]

B. Data Collection and Preprocessing

- remove non-UDP traffic
- isolate collections of queries and responses that correspond to a single trusted or untrusted source address
- count parameters in individual queries and responses within regular sample intervals
- normalise counts in each sample

C. Feature Engineering

D. Validation Process

Validation process (hold-out, k-fold, ...)

No results

[5]Nominet was acquired by Akamai in November 2017

[6]PCAP, named after the C library `libpcap`, is the file format used by the `tcpdump` utility.

No code

Formulas can be included as long as they are not specific to a particular programming language.

V. Evaluation

Results of the process I applied

Can include a paragraph describing what languages, packages and libraries were used.

Possibilities: accuracy measures, graphs showing tuning process, tables and graphs comparing different approaches, tuned parameter ranges and selected values.

No code.

VI. Conclusion

Short summary of the paper/report

Should include problem description, how I solved it and the main results.

VII. Colophon

This document has been written in R Markdown[7]; the code used to produce the output included in this document is consequently included with the document source[8]. The production of this document in IEEEtran style from R Markdown was informed by a pseudonomymously-attributed community project[9].

References

[1] "Domain names - concepts and facilities," RFC 1034, Nov. 1987. [Online]. Available: https://rfc-editor.org/rfc/rfc1034.txt

[2] "Domain names - implementation and specification," RFC 1035, Nov. 1987. [Online]. Available: https://rfc-editor.org/rfc/rfc1035.txt

[3] "Transmission Control Protocol," RFC 793, Sep. 1981. [Online]. Available: https://rfc-editor.org/rfc/rfc793.txt

[4] "User Datagram Protocol," RFC 768, Aug. 1980. [Online]. Available: https://rfc-editor.org/rfc/rfc768.txt

[5] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. E. Hoffman, "Specification for DNS over Transport Layer Security (TLS)," RFC 7858, May 2016. [Online]. Available: https://rfc-editor.org/rfc/rfc7858.txt

[6] P. E. Hoffman and P. McManus, "DNS Queries over HTTPS (DoH)," RFC 8484, Oct. 2018. [Online]. Available: https://rfc-editor.org/rfc/rfc8484.txt

[7] C. Huitema, M. Shore, A. Mankin, S. Dickinson, and J. Iyengar, "Specification of DNS over Dedicated QUIC Connections," Internet Engineering Task Force, Internet-Draft draft-huitema-quic-dnsoquic-05, Jun. 2018, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-huitema-quic-dnsoquic-05

[8] F. Neves and J. Damas, "Preventing Use of Recursive Nameservers in Reflector Attacks," RFC 5358, Oct. 2008. [Online]. Available: https://rfc-editor.org/rfc/rfc5358.txt

[9] D. J. C. Klensin, "Simple Mail Transfer Protocol," RFC 5321, Oct. 2008. [Online]. Available: https://rfc-editor.org/rfc/rfc5321.txt

[10] K. E. Lindqvist and J. Abley, "Operation of Anycast Services," RFC 4786, Dec. 2006. [Online]. Available: https://rfc-editor.org/rfc/rfc4786.txt

[11] K.-B. Duan and S. S. Keerthi, "Which is the best multiclass svm method? an empirical study," in *Multiple Classifier Systems*, N. C. Oza, R. Polikar, J. Kittler, and F. Roli, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 278–285.

[12] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324

[13] Y. Yuzifovich, H. Liu, A. Sarychev, and A. Asiaee, "Augmented Intelligence to Scale Humans Fighting Botnets," in *Proceedings of botconf 2017*, Dec. 2017. [Online]. Available: https://www.botconf.eu/2017/augmented-intelligence-to-scale-humans-fighting-botnets/

[14] Y. Yuzifovich, "What's lurking in core domains," in *Proceedings of OARC 27*, Sep. 2017. [Online]. Available: https://indico.dns-oarc.net/event/27/contributions/456/

[15] J. Qiao. (2018, Nov.) Detecting resolvers at .nz. [Online]. Available: https://blog.nzrs.net.nz/detecting-resolvers-at-nz/

[16] J. Qiao and S. Castro, "Resolver detection using machine learning," in *Proceedings of OARC 29*, Oct. 2018. [Online]. Available: https://indico.dns-oarc.net/event/29/contributions/655/

[17] M. Sammour, B. Hussin, and M. F. I. Othman, "Comparative analysis for detecting dns tunneling using machine learning techniques," *International Journal of Applied Engineering Research*, 2017. [Online]. Available: https://www.ripublication.com/ijaer17/ijaerv12n22_137.pdf

[7] https://rmarkdown.rstudio.com

[8] https://github.com/ableyjoe/uwo-mesc/tree/master/ECE-9603A-001-GF18/project

[9] https://github.com/mathematicalcoffee/IEEEtran-rmarkdown