Predentifier: Detecting Botnet C&C Domains From Passive DNS Data

Tilman Frosch, Marc Kührer, Thorsten Holz
Horst Görtz Institute (HGI), Ruhr-University Bochum, Germany

Abstract

The Domain Name System (DNS) is mainly used for benign and legitimate Internet activities. Nevertheless, it also facilitates malicious intentions. Domain names have started to play an increasingly important role in the Command and Control (C&C) infrastructure of botnets. These domains can be added to blocklists or taken down, yet attackers can simply evade the countermeasures by creating hundreds of new domains every day.

To detect C&C domains at an early stage, we propose a framework called Predentifier that combines a host's DNS configuration properties with secondary data to derive a set of distinctive features that can be used to describe the behavior of a host. We employ methods of statistical learning to determine whether a domain belongs to a C&C server or if it is benign. We further show that it is possible to leverage passive DNS data to identify C&C domains without infringing on employment or customer rights.

1 Introduction

In recent years, domain names have started to play an increasingly important role in *Command and Control* (C&C) mechanisms of *botnets*, i.e., networks of compromised machines under the control of an attacker (often called *botmaster*) [6, 8, 15]. Botnets are responsible for some of the major problems on the Internet: they are used to propagate spam and to steal of banking credentials and accounts to a variety of online services, among other criminal activities like *Distributed Denial of Service* (DDoS) attacks [14]. Many botmasters today maintain control over their criminal assets by using DNS-based C&C structures to prevent efforts to take down the botnet. A timely identification of C&C domains can allow for the detection of a botnet even before it is put to use on a large scale. We introduce Predentifier to effectively identify C&C domains at an early stage – without infringing on employment or customer rights. The approach combines DNS configuration properties of a host with secondary data like WHOIS and geolocation information. We derive a set of 14 distinctive features and employ methods of statistical learning to decide with a high confidence, whether a domain is used for C&C or is affiliated with a benign, legitimate participant on the Internet.

2 Related Work

The idea of performing passive DNS replication to detect malware was introduced by Weimer [16] in 2005. Zdrnja et al. [19] adopted this idea and proposed a passive DNS system to trace hosts associated to botnet C&C servers. To identify hosts and domains which are participating in Fast-Flux networks, Holz et al. [11] analyzed DNS records aggregated with further information like *Autonomous System Numbers* (ASN) and geolocation data. Yadav et al. [18] proposed an approach to detect domain fluxes in DNS traffic by identifying domain names which have been generated algorithmically. Felegyhazi et al. [7] investigated DNS properties and registration information to explore the potential of proactive domain blacklisting. Antonakakis et al. [3] presented the dynamic reputation system Notos based on DNS and secondary data provided by honeynets and malware analysis services, which analyzes network and zone features to describe characteristics of domains. In a second paper, Antonakakis et al. [4] introduced a system which attempts to detect malicious domains by analyzing passive DNS data gathered at authoritative nameservers and top-level domain servers. Bilge et al. [5] proposed the architecture Exposure which analyzes passive DNS data to automatically distinguish between malicious and benign domains. The authors introduce time-based and DNS-based features.

Besides implementations using (passive) DNS data, other approaches have been published focusing on lexical and host-based features to distinguish between benign and malicious hosts [12, 10].

3 Motivation: DNS Features of Botnet Domains

The intuition of our approach is that the requirements towards hosts used for controlling botnets differ from the requirements a user has towards a server providing benign content. These requirements are also reflected in the DNS configuration. While a well-established site rarely changes the IP address(es) it resolves to, the maintainer of a botnet C&C server may suddenly need to change an `A`-record in order to maintain continuous control over the bots. Be it because the legitimate owner of a compromised host used as C&C server disconnects his system from the Internet, or that a server, legitimately acquired by the botherder, is seized by law enforcement officials in an attempt to shut down the botnet. As such a configuration change needs to propagate quickly, this policy is reflected in a lower TTL value in the DNS configuration and may, for example, also be reflected in the *refresh* value that determines the time between two zone transfers requested from secondary nameservers.

Redundancy is defined differently for benign and C&C domains. This is reflected in the amount of IP addresses a domain resolves to. A benign site can balance load by simply resolving a hostname to a set of IP addresses in a round-robin fashion. High-traffic sites are also often hosted on *Content Delivery Networks* (CDN) and behave differently compared to ordinary hosts offering benign content [11]. Botnet C&C servers will receive fewer requests than high-traffic sites, thus do not need load-balancing in the same way. What the botherder needs instead is a way to mitigate takedowns of C&C servers currently in use. This can be achieved by changing the `A`-record for this hostname. As a consequence, the average amount of IP addresses being resolved for one malicious hostname at a time may be lower than the amount of addresses seen in the context of a benign domain. However, this assumption highly depends on the data set and whether there are any active *fast flux* [17] domains in the set of malicious domains.

A hypothesis in the context of WHOIS data implies that benign domains tend to be older than domains used for botnet C&C purposes [5]. While analyzing the registration dates of the domains in our data sets, we indeed found that on average the age of C&C domains is significantly lower.

Other properties that are expected to differ for benign and malicious domains are the geographic locations of the IP addresses being resolved from one hostname and their locations within the network as reflected by the ASNs. Servers used for benign purposes and addressed by the same domain are often located in the same country. This still holds true for massively loadbalanced setups and CDNs. Further on, a legitimate business might rather choose to have all its hosting services provided by one organization which will result in only a few ASNs or even only one. In contrast, a botmaster tends to act opportunistic and use any host that is available and fits his needs. A widely distributed architecture does not proof disadvantageous – on the contrary, spreading C&C operations over more than one legislation may increase the resilience against takedowns by law enforcement.

4 System Design

To generate detection models for C&C domains based on the assumptions and observations in the previous section, Predentifier applies 14 distinctive features derived from passive DNS, WHOIS, and geolocation data.

4.1 System Outline

Figure 1 outlines the architecture of Predentifier. In the offline phase, one of the core components of our analysis system is the database storing labeled benign and malicious domains. We generate a training set $\mathcal{S}_{\mathcal{TRAIN}}$ by randomly choosing labeled benign and malicious domains from the database. We then attribute features to the individual hostnames, based on passive DNS, WHOIS, and geolocation information. From these information we create a detection model to be used by the classifier. In a next step, we choose a test set $\mathcal{S}_{\mathcal{TEST}}$ of domains and also attribute features to each element of the set. However, the labels Y_i' are removed from the set. We classify all elements of $\mathcal{S}_{\mathcal{TEST}}$ and compare the inferred class Y_i with the original labels Y_i' of the test set to determine true and false positives. When used as an online system, several passive DNS sensors, deployed in various networks, collect DNS answers from the network traffic. Again, features are attributed to these hostnames, which are then classified based on the previously built model.

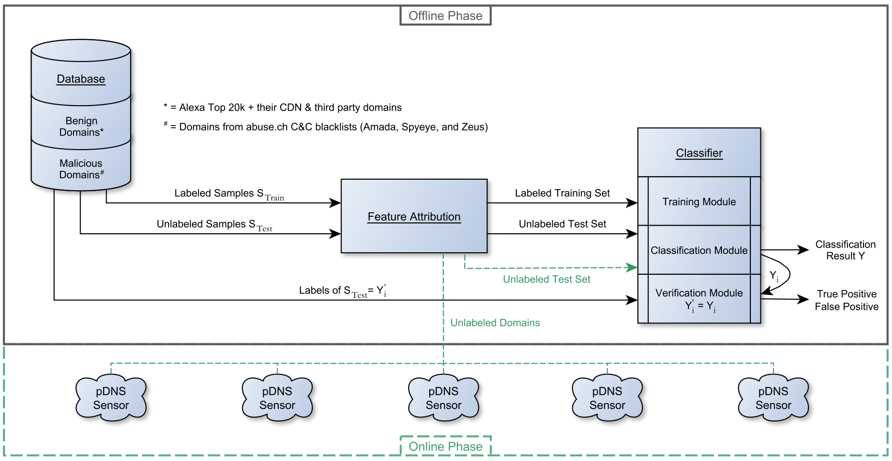

Figure 1: System architecture of Predentifier

4.2 Data Acquisition

A very common approach to acquire a set of benign hostnames is to use the TOP 20.000 domains from the Alexa traffic ranking [2]. Most of these sites, however, make use of CDNs and other sites providing third-party content that consequently are equally highly frequented as the site they serve content to. Some of these may exhibit similar behavior than malicious domains, e.g., in terms of zone configuration. We include the *Fully Qualified Domain Name* (FQDN) of each host that provided content to these sites, so the training and test sets also contain these domains and the model is created accordingly. Additionally to the 20.000

primary domains, we found 40,881 domains that are used to provide third-party content to the high-traffic domains or were part of a CDN. We verified that none of the resulting 60,881 domains was found on any of the publicly available C&C blacklists.

As *malicious* domain set we use a subset of domains accumulated from three C&C server blacklists provided by abuse.ch [1]: their Malware Database C&C blocklist (AMaDa) as well as the dedicated ZeuS and SpyEye trackers.

4.3 Classifier

For the classification process we use *k-Nearest Neighbor* (kNN). kNN is one of the most straight-forward supervised learning methods. The simplified basic assumption is that samples defined by an n-dimensional vector which are closest in this vector space must be similar, i.e., it determines the decision boundary locally. The parameter k describes how many neighboring points should be taken into account. Equation 1 shows the k-nearest neighbor fit Y for a classification for an unclassified sample x, where $N_k(x)$ is the neighborhood of x defined by the k-closest points in the training set [9].

$$Y(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i \tag{1}$$

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{2}$$

A common metric to define closeness is the Euclidean distance. Equation 2 shows the Euclidean distance d of two vectors x, y consisting of values x_i, y_i.

A classification with $k = 1$ is not very robust as any new sample x is simply assigned to the class of the nearest element in the training set. kNN with $k > 1$ is more robust as it assigns samples to the majority class of their k-closest neighbors [13]. A low value k will introduce more noise intro the results. On the other hand, a high value of k renders kNN computationally more expensive. It also conflicts the basic idea behind kNN, i.e., that points, which are near to each other, are more likely to belong to a similar class than points with a higher distance.

4.4 Feature Selection

As already outlined in Section 3, properties observed from passive DNS data and other sources reveal information to describe either malignance or benignity of a domain. However, the choice among possible features is strongly determined by data availability and level of disaggregation: the passive DNS data available to us is aggregated with respect to the temporal dimension. Thus, we are not able to analyze patterns of repeated usage, daily similarity in when and how often a domain is accessed and similar features. Instead, we introduce a number of other features that evolved from the available data.

#	Feature Name	Description
1	digitratio	Number of digits compared to length of domain
2	consonantratio$^{\text{novel}}$	Number of consonants compared to length of domain
3	consonantvocalratio$^{\text{novel}}$	Number of consonants compared to amount of vocals
4	ipcount	Number of IP addresses the domain resolves to
5	ttl_max$^{\text{novel}}$	Maximum TTL during observation (in sec.)
6	ttl_min$^{\text{novel}}$	Minimum TTL during observation (in sec.)
7	ttl_diff$^{\text{novel}}$	$Max.\ TTL - min.\ TTL$ during observation (in sec.)
8	soa_sn_changes$^{\text{novel}}$	Number of SOA S/N changes during observation
9	exp_time_min$^{\text{novel}}$	Minimum expiry time of domain's zone (in sec.)
10	rtry_time_min	Minimum retry time of domain's zone (in sec.)
11	rfrsh_time_min$^{\text{novel}}$	Minimum refresh time of domain's zone (in sec.)
12	countrycount	Number of countries the domain is served from
13	asncount	Number of ASNs the domain is served from
14	domainage$^{\text{novel}}$	*last_seen date* − *domain creation date* (in days)

Table 1: Features utilized by PREDENTIFIER

As shown in Table 1, we identified a set of 14 distinct features of which nine features have, to the best of our knowledge, never been tested in previous work and can be considered novel. Fig. 2 outlines the arithmetic average and standard deviation for each feature normalized to the *global average* of each feature, i.e., the arithmetic average of the feature value as found in the combination of both the benign and the malicious set. Normalization provides a better overview of the differences between malicious and benign domains and eases the interpretation of the different features as the features vary widely in scale.

Lexical Features: Ma et al. [12] justify lexical features based on the observation that malicious URLs tend to look differently compared to benign URLs. Our approach does not deal with URLs as a whole but with a subset of the URL: the domain. The argument is also confident for the domain since it may look different, especially for domains resulting from a *Domain Generation Algorithm* (DGA). Our lexical features include the number of digits compared to length of the domain, the amount of consonants compared to length, and the number of consonants compared to the amount of vocals, based on the assumption that these ratios differ between words of a spoken language and (randomly) generated strings.

DNS Answer-based Features: Feature 4 consists of the number of IP addresses a domain is resolved to during the observation period. The intuition for

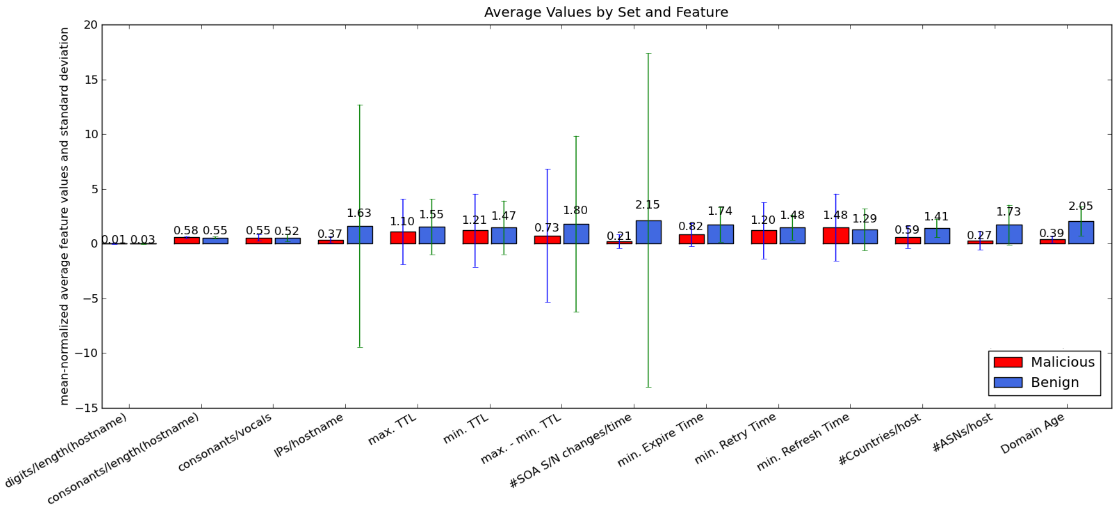

Figure 2: Mean-normalized average feature values and standard deviation

using this feature is that legitimate websites receiving significant traffic tend to use several hosts in parallel to balance load. A round-robin resolution appears as A-records of one domain resolved to different IP addresses, where each database entry exhibits a *last_seen* timestamp of similar age. This still holds true for CDNs, although only for a given geographic and network-topologic location. The amount of hosts that deliver the same content to a user in a given area via a given ISP varies widely depending on the time of day [11]. However, aggregated over an observation period of seven to 30 days, the amount of individual active IP addresses stays relatively constant for a given website. Consequently, CDN hosts and traditionally loadbalanced sites can be assumed to behave similar with respect to feature 4. The number of IP addresses per domain, however, varies highly depending on the actual domain name. For a botmaster it is crucial that only few of the C&C servers appear in public. Thus, a domain used in a C&C context is more likely to be resolved to only few IP addresses at a time. The distinctiveness of this feature alone is of course limited, as small benign websites also use just one IP address.

IP Address-based Features: IP address-based features are attributed to a domain indirectly, such as the number of different countries and ASs a client is served from. Sample data indicates that the amount of different countries and ASs a domain is affiliated with is higher for benign domains. Yet, botmaster may take advantage of heterogeneous jurisdictions by spreading C&C servers over several countries or using any host available. This observation is also shared by Bilge et al. [5] with regard to geolocation and by Antonakakis et al. [4] with regard to ASs.

Some high-traffic websites might do load-balancing multi-nationally, but most will simply make use of CDNs to reduce load and latency. It follows that again the large majority of benign domains is being served from one country and one AS. However, the passive DNS database is fed by sensors at a variety of locations which may each see a different set of IP addresses associated with a domain. As a consequence, the amount of different ASNs and countries associated with a domain is on average slightly higher for benign domains. However, standard deviation indicates that these features can be volatile, as there exist many benign hostnames pointing to exactly one IP address located in one country.

Zone-based Features: These features are derived from SOA resource records and cover the minimum and maximum TTL, the difference between those values as well as the number of serial number changes for the respective SOA record, and the minimal values for expiry time, retry time, and refresh time.

A lower TTL value allows for a higher flexibility as it reduces caching time on the client side, e.g., for hostname ↔ IP address relations. For a benign domain, the DNS configuration for an individual host will seldom change. Under these premises, a high TTL reduces the load on the domain name system while the slow propagation of changes is tolerable. In a C&C setting, the maintainer may be confronted with less foreseeable events that make it necessary to update a record faster. The feature, however, is not distinctive on its own: a very low TTL is also used for load-balancing via round-robin DNS [5]. Besides using the maximum and the minimum of a domains's TTL, we record the difference between both TTLs which indicates a change in zone behavior.

Refresh time, retry time, and expiry time are similar indicators for desired flexibility but on a nameserver infrastructure level. The refresh time is the equivalent of TTL for SOA records, i.e., the time a secondary nameserver will wait before querying the primary server for changes. The retry time determines how long a host will wait before retrying after a failed zone transfer. 80% of the domains in the malicious set are configured with a $retry\ time \leq 1800\ seconds$ while the same is true for only about 21% of the benign samples. The tendency towards lower retry times observed with malicious zones can again be interpreted as an attempt to make the complete setup more resilient against failures and offer the possibility for faster recovery. The expiry time determines how long a secondary nameserver will continue to try to complete a zone transfer from a primary server. The average expiry time in the benign random set is by factor 2.13 higher than in the malicious set, and the standard deviation of the non-normalized values indicates that the respective value intervals do not overlap.

The amount of SOA serial number changes indicates how often a zone is updated during the observation period. Our data indicates that this differs for benign and malicious zones, thus we use the amount of serial number change as a feature.

WHOIS-based Features: The registration date of a domain is provided by many registries in the WHOIS answer and can be used to calculate the age of a domain. Bilge et al. [5] observe that malicious domains often have a short life span. It follows that one can expect to observe a lower age in this group of domains when compared to benign domain names that are often part of a well-established infrastructure. While we have observed domains blacklisted for C&C with a domain age above 3000 days, the averages of the benign and malicious random set vary in the scale of years. The average age of a benign domain in the sample is 2276 days, while the average age of a malicious domain is 416 days with a median of 306 days (2086 days for the benign set). Also, only 3% of the random benign domains are up to a year old or younger while this is true for more than 62% of the malicious domains.

5 Evaluation

We first test the soundness of the approach by performing 10-fold cross validation on different sets of hosts. We then test the effectiveness in a realistic scenario. In a real-world scenario, Predentifier would be required to train on data observed in the past, i.e., the time period $[t.now - n, ..., t.now]$ and use the resulting model to correctly identify C&C servers from data observed in the time period $[t.now + 1, ..., t.now + m]$, where $t.now$ describes the time being, n is a value indicating the end of the observation period in the past, and m is the time period after that the classifier is re-trained. We simulate this scenario by using data from domains observed between a date in the past d_p and a date even further in the past $d_p - n$ to compile a training set and test it against data observed from domain names first seen at the time $d_p + 1$ until the day we performed the evaluation. In a next step, we evaluate the contribution our novel features offer to the classification results.

5.1 Evaluation Data

In the following, we refer to the 60,881 domains derived from the TOP 20,000 domains in the Alexa traffic ranking as the *global benign set* $\mathcal{B}$. Furthermore, we refer to the set of samples associated with any entry in any of the abuse.ch blacklists as the *global malicious set* $\mathcal{M}$. We refer to a labeled set of samples as *training set* $\mathcal{S}_{\mathcal{TRAIN}}$. When a set is referred to as *test set* $\mathcal{S}_{\mathcal{TEST}}$ it is an unlabeled set of samples. The classification process assigns a sample $x \in \mathcal{S}_{\mathcal{TEST}}$ either to the benign or the malicious class. We calculate the *false positive rate* as the percentage of benign samples that is falsely classified as malicious and the *detection rate* or *true positive rate* as the percentage of malicious samples that is correctly classified.

5.2 Cross Validation

N-fold cross validation is a technique frequently used to evaluate the effectiveness of a proposed detection mechanism.Applying this technique splits the data into N random partitions. $N-1$ partitions are used to train the classification algorithm, and the resulting model is then tested on the remaining partition. The N resulting detection and false positive rates are then averaged.

We use 10-fold cross validation to evaluate the detection accuracy on data observed during a 30-day period. In the context of this evaluation, we also determine which value k in kNN yields the best classification results. For every test, the cross validation set is the same and consists of identical partitions for every run, i.e., we randomize and partition the set once and keep this configuration for every run. Thus, the only parameter that varies between the different cross validations is the value of k.

The sets used for the cross validation contained the following amount of samples: $|\mathcal{S}_{\mathcal{TRAIN}}| = 17487$ and $|\mathcal{S}_{\mathcal{TEST}}| = 1943$. These two sets consist of a total of 17,386 benign and 2,044 malicious domains. We evaluated false positive and detection rate for classifications with kNN, where $k \in [1, 2, ..., 15, 50, 150, 1500]$. On this data set a downward trend can be observed for $k > 2$.

The best detection rate of 93.6% in combination with a low false positive rate of 0.5% was achieved with $k = 2$. The results from the 10-fold cross validation on these two different sets show that the approach is indeed fit to detect botnet C&C servers from passive DNS information in combination with secondary data. The test on the second data set shows that good results can be achieved both when classifying hosts, whose activity has been detected temporally near to each other and also when classifying hosts that may last have been observed in a temporal distance of up to 30 days.

5.3 Temporally Disjoint Data Sets

This kind of evaluation can be considered a reality check for our approach. A system based on this approach should be able to successfully detect C&C servers it observes in the future based on training data observed in the past. In order to represent this scenario in a realistic manner, we randomly choose ten training sets, each consisting of 500 benign and 500 malicious samples from domains observed between 2011-12-07 and 2012-01-15. We use the feature values determined for these samples to calculate global average values for each feature. As test set we use all domains observed between 2012-01-16 and 2012-02-07 that fulfill the requirement of having at least five determinable host-based features. From this data we randomly choose ten training sets and ten identical test sets. The latter consist of 1,185 benign and 822 malicious samples.We run kNN using each of the training sets as input and test the resulting model on the test set. The evaluation result is reported as the average detection and false positive rate calculated over these ten runs. In a real-world application, one would also draw ten random sets and automatically evaluate these training sets on disjoint test sets from the same period. While the classification results vary only slightly between the ten different training sets, one would choose the model that yields the best results. Thus, the reported average values can be considered a conservative estimate.

We repeat this procedure for $k = (1, 2, ..., 22)$. As already observed in the cross validation, the best results can be achieved with a choice of $k = 2$: a detection rate of $\approx 94.2\%$ and false positive rate of $\approx 8.7\%$.

5.4 Feature Effectiveness

Approaches like Exposure [5] that rely on a substantial amount of time-based features make it necessary to store each and every DNS answer observed in a network. Due to storage constraints, we store the data of each DNS answer containing an identical resource record only once. While this reduces the space complexity of our approach, it renders the whole class of time-based features used by Bilge et al. unavailable to us. We faced a similar challenge with respect to features used in other approaches. Thus, we derived features from the available data to compensate for the unavailable features.

We also utilize the data used in Section 5.3 to evaluate the contribution each features makes to the classification process. Again, we use ten randomly drawn training sets and give the resulting detection and false positive rates as the average from these classifications. Additionally, we calculate the standard deviation as a measure of volatility of the classification accuracy with respect to the training set. As $k = 2$ has been shown to yield the best results, we compare classification results using kNN with $k = 2$. The *baseline feature set* contains the features 1, 4, 10, 12, and 13. This is the intersecting set of the feature sets used in previous work with the feature set that can be derived from the data available to us. Classification using only these features results in a detection rate of 58.86% and a false positive rate of 7.21%. However, the standard deviation of the detection rate is $\approx 27.41\%$ and 2.72% for the false positive rate. This shows that the classification result is highly dependent on accidentally drawing a "good" random training set. We use this configuration as baseline to assess the contribution of each individual feature, i.e., we examine how the classification results change when we selectively add each novel feature or each group of features derived from the same kind of data. For instance, the bars labeled with *+consonantratio (2)* depict the detection and false positive rate of a classification using the baseline feature set plus feature 2, which is *consonantratio*. All results are shown in Fig. 3.

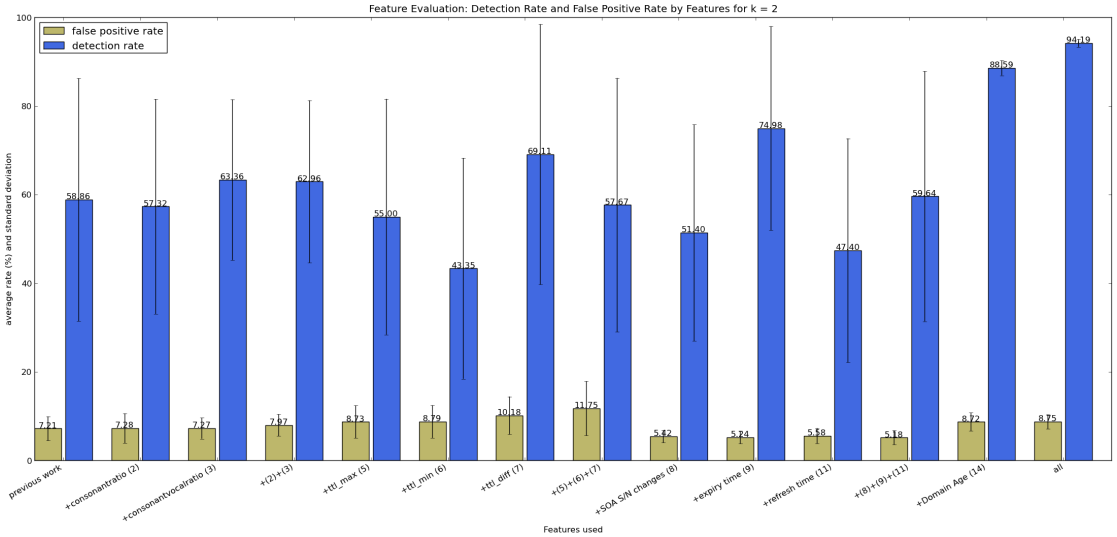

Figure 3: Detection rate, false positive rate, and respective standard deviation by features used

The combination of all novel features added to the baseline feature set has a significant positive effect on the detection rate by increasing it from 58.86% to 94.19%. However, the newly introduced features also slightly increase the false positive rate by 1.54%. Yet, compared to the baseline set, these complete features are stable: when evaluating the same test set with ten different models based on ten randomly chosen training sets, the standard deviation is 1.64% for the false positive rate and only 0.86% for the detection rate while in the baseline feature set it is 2.72% and 27.41%, respectively. In summary, our features significantly improve the classification results.

6 Conclusion

We showed how data acquired via passive DNS replication and additional data like WHOIS and geolocation information can be used to identify botnet C&C domains. Based on our observations, we proposed 14 effective features to determine domain characteristics from sparse data. To the best of our knowledge, nine of these features have never been used before – thus provide an enlargement of the available feature choices. The proposed features can be divided into lexical features that are derived from the domain name itself, DNS-answer-based features that are directly derived from resource records, and IP address-based features like the host's geographic location and its location within the Internet's topology. Taking advantage of these features, we developed an approach to detect botnet C&C

servers using machine learning methods. We tested the soundness of our approach on temporally disjoint training and test sets, a scenario as similar as possible to a real-life application environment of such a system. We found that our approach maintains a high detection rate of 94.2% and a relatively low false positive rate of 8.75%. Passive DNS replication is a very privacy-preserving technology, as only DNS answers are stored – information that is publicly available within the domain name system. At no point we make use of personally identifiable information. Our approach is thus fit to be used in a privacy-sensitive context.

Acknowledgements

This work has been supported by the German Ministry of Economic Affairs and Energy of the State of North Rhine-Westphalia under Grant 290077302 (*MAKE*) and the Federal Ministry of Education and Research under BMBF Grant 01BY1111 (*MoBE*). We would like to thank Aaron Kaplan and the CERT.at team for providing us access to their passive DNS database.

References

[1] abuse.ch. AMaDa, SpyEye and ZeuS C&C Domain Blocklists. `http://www.abuse.ch/`, 2012.

[2] Alexa. Top 1,000,000 sites. `http://www.alexa.com/topsites`, July 2011.

[3] Manos Antonakakis, Roberto Perdisci, David Dagon, Wenke Lee, and Nick Feamster. Building a dynamic reputation system for DNS. In *USENIX Security Symposium*, 2010.

[4] Manos Antonakakis, Roberto Perdisci, Wenke Lee, Nikolaos Vasiloglou, and David Dagon. Detecting malware domains at the upper DNS hierarchy. In *USENIX Security Symposium*, 2011.

[5] Leyla Bilge, Engin Kirda, Christopher Kruegel, and Marco Balduzzi. EXPOSURE: finding malicious domains using passive DNS analysis. In *Network and Distributed System Security Symposium*, 2011.

[6] Evan Cooke, Farnam Jahanian, and Danny McPherson. The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets. In *USENIX Workshop on Steps to Reducing Unwanted Traffic on the Internet(SRUTI)*, 2005.

[7] Mark Felegyhazi, Christian Kreibich, and Vern Paxson. On the potential of proactive domain blacklisting. In *Proc. of the Third USENIX Workshop on Large-scale Exploits and Emergent Threats (LEET)*, 2010.

[8] Felix C. Freiling, Thorsten Holz, and Georg Wicherski. Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks. In *European Symposium on Research in Computer Security (ESORICS)*, 2005.

[9] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, 2nd edition, 2009.

[10] Yuanchen He, Zhenyu Zhong, Sven Krasser, and Yuchun Tang. Mining DNS for malicious domain registrations. In *Proc. of the 6th International Conference on Collaborative Computing: Networking, Applications and Worksharing*, 2010.

[11] Thorsten Holz, Christian Gorecki, Konrad Rieck, and Felix C Freiling. Measuring and detecting Fast-Flux service networks. In *Network & Distributed System Security (NDSS)*, 2008.

[12] Justin Ma, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. Beyond blacklists: Learning to detect malicious web sites from suspicious URLs. In *Proc. of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009.

[13] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval.* Cambridge University Press, 2008.

[14] Jelena Mirkovic and Peter L. Reiher. A taxonomy of DDoS attack and DDoS defense mechanisms. *Computer Communication Review*, 34(2), 2004.

[15] Moheeb Abu Rajab, Jay Zarfoss, Fabian Monrose, and Andreas Terzis. A Multifaceted Approach to Understanding the Botnet Phenomenon. In *Internet Measurement Conference (IMC)*, 2006.

[16] Florian Weimer. Passive DNS replication. In *17th Annual FIRST Conference on Computer Security*, 2005.

[17] Sandeep Yadav and A.L. Narasimha Reddy. Winning with DNS failures: Strategies for faster botnet detection. In *Proc. of the 7th International ICST Conference on Security and Privacy in Communication Networks*, 2011.

[18] Sandeep Yadav, Ashwath Kumar Krishna Reddy, A. L. Narasimha Reddy, and Supranamaya Ranjan. Detecting algorithmically generated malicious domain names. In *Proc. of the 10th annual conference on Internet measurement*, IMC '10, 2010.

[19] Bojan Zdrnja, Nevil Brownlee, and Duane Wessels. DNSParse. `https://dnsparse.insec.auckland.ac.nz/dns/technical.htm`, 2007.