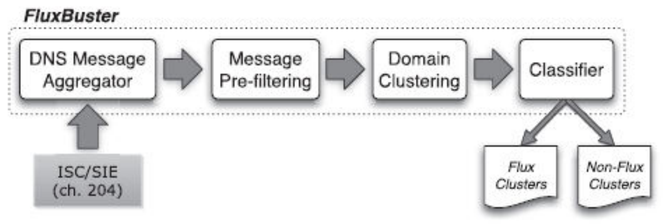
Chapter 3

Proposed System : FluxBuster

This chapter describes how FluxBuster works in detail.Lets start with a system overview.

3.1 System Overview



High-level system overview.

The Figure above shows a high-level overview of FluxBuster. FluxBuster receives in input a stream of DNS messages collected through the ISC/SIE infrastructure, as shown.Each collected DNS message provides information regarding the mapping of a domain name d to a set of resolved IP addresses R, as observed from the ISC/SIE DNS sensors distributed across the Internet.

The DNS Message Aggregator module aggregates all DNS messages regarding a domain d into a higher level DNS message Q(d) that contains all information about d observed during a given interval of time Δ. Each Q(d)includes information such as how many DNS queries to d were observed during Δ, the aggregated set of all resolved IP addresses pointed by d, their average time-to-live (TTL), etc. The length of Δ should be at least a few hours, so that enough information can be collected (e.g., several resolved IPs) for each flux domain name.

The Message Prefiltering module analyzes each aggregated DNS message Q(d) and filters out those messages related to domains d that are very unlikely to be flux domains. In practice this module performs data volume reduction, and aims to reduce the computational cost of the more fine-grained analysis performed by the following modules.The Message Prefiltering module is implemented using a set of very conservative filtering rules. The consequence is that at this stage FluxBuster accepts all the flux domains as well as several nonflux domains, such as domains related to CDNs, as candidate flux domains for further analysis.

The set of domains D ={ d1; d2; . . . ; dn } and the related set of aggregated messages Q(d1),Q(d2), . . .,Q(dn) that pass the prefiltering step are then sent to the Domain Clustering module. This module partitions the set of domains D into a set of domain clusters C = { C1; C2; . . . ; Cm }, with D = ∪Ci. Each of these clusters contains domains that are related to each other, because they share a nonnegligible percentage of resolved IPs. In practice, these clusters of domains and the related sets of IP addresses represent candidate flux networks.

Because of the conservative filtering approach taken by the Message Prefiltering module, not all these candidate flux networks are actually flux. Therefore, in order to distinguish between flux and nonflux networks, FluxBuster employs a Classifier module based on a supervised statistical algorithm that has been previously trained on examples of both flux and nonflux networks. The Classifier module processes the clusters produced by the Domain Clustering module that include a number of IP addresses larger than a predefined threshold Δ (e.g., $\Delta = 30$), and labels them as either flux or nonflux.The threshold Δ is used first of all because clusters with very few IP addresses are extremely unlikely to represent active flux networks, and because sufficiently large and representative sample set of IP addresses are needed for each candidate flux network in order to compute some of the statistical features described later. Therefore, clusters that contain a number of IP addresses lower than Δ cannot be reliably labeled as flux, and are simply discarded.

3.2 DNS Message Aggregator

FluxBuster receives in input a stream of DNS messages as provided by the ISC/SIE framework. ISC/SIE collects raw DNS query/response messages from a large number of RDNS sensors, and rebroadcasts these DNS messages in a deduplicated fashion.For example, assume that there are three RDNS sensors S1, S2, and S3 that have reported a DNS query/response message regarding a domain name d to ISC/SIE. Suppose that S1 reported the mapping of domain d to three IP addresses, { IP1, IP2, IP3}, S2 reported the mapping of d to two addresses {IP1, IP4}, and S3 reported the mapping of d to one address {IP5}. These raw messages will be combined within the ISC/SIE framework into a deduplicated message stating that d maps to {IP1, IP2, IP3, IP4, IP5}. In other words, the deduplication process aggregates raw DNS query/response messages received within a certain time window T into a single message that summarizes the domain-to-IP mappings observed during T from multiple sensors. Because of the implementation of the ISC/SIE deduplication system, the value of T varies with time, and it is in the order of a few hours (e.g., two to four hours).

FluxBuster performs a further aggregation of the messages that are received from ISC/SIE within a period Δ equal to 12 hours. The aggregated message can be represented as a tuple Q(d) = { d, t1, t2, R, τ, c }, where d is a given domain name, t1 and t2 represent the timestamp of the first and last DNS query/response messages regarding d that were aggregated into Q(d), respectively, R is the set of all IP addresses resolved during the period (t2 - t1),τ is the average time-to-live of the DNS responses, and c is the number of raw DNS query/response messages aggregated by Q(d).

3.3 Characteristics of Flux Domain Names

Fast-flux domains are characterized by the following main characteristics:
1. short time-to-live;
2. high frequency of change of the set of resolved IPs (i.e., the flux agents) returned at each query;
3. the overall set of resolved IPs obtained by querying the same domain name over time is often very large; and
4. the resolved IPs are scattered across many different networks.

Some legitimate services, such as legitimate CDNs, NTP server pools, IRC server pools, etc., are served through sets of domain names that share some similarities with fast-flux domains.Although the value of each individual characteristic may not allow us to precisely identify malicious flux domains and distinguish them from legitimate domains, combining multiple measures based on the above characteristics enables accurate detection of flux domains and networks.

3.4 Message Prefiltering

The Message Prefiltering module performs data volume reduction by discarding domain names that are very unlikely to be part of a flux network.Given an aggregated DNS message Q(d) related to a domain d, Q(d) is accepted by the filter, i.e., it is considered for further analysis, if all the following rules are matched:
1. $\tau \leq \theta'_{ttl}$,
2. R $\geq \theta_R$ OR $\tau \leq \theta''_{ttl}$, and
3. div(R)$\leq \theta_{div}$,
where θ''_{ttl}, θ'_{ttl}, θ_R, and θ_{div} are suitably chosen thresholds (with $\theta''_{ttl} < \theta'_{ttl}$). The function div(R) computes the diversity of the resolved IP set R, and it is computed as $div(R) = \frac{|P|}{|R|}$, where P is the set of all the /16 IP prefixes in R. P is computed by considering each IP in R, and extracting its /16 prefix (e.g., the /16 prefix of 128.192.76.177 is 128.192). Thus it follows that $|P|$ is the number of distinct IP prefixes in R, and a large value of div(R) is associated to a set of resolved IPs that are scattered across many different / 16 networks.

To make sure that the Message Prefiltering module does not discard flux domains,the filtering thresholds are set as follows: $\theta'_{ttl} = 3$ hours, $\theta_R = 3$, $\theta'_{ttl} = 30$ seconds, and $\theta_{div} = \frac{1}{3}$. Therefore, only domains with a very large TTL, very low number of resolved IPs, and a low value of diversity of the IP set will be discarded.It is also worth noting that the threshold θ''_{ttl} is used to avoid discarding those domains that resolve to a very small set of IPs, and at the same time are characterized by a very short TTL. This rule has been introduced because some attackers setup their flux domains to resolve to only one flux agent (i.e., one IP address) per query. However, in this case, the TTL is often set to zero (or at most a few seconds) to guarantee that the flux agents IP is not cached by the local DNS resolver for too long, and the next time a user clicks on the same domain she will obtain a fresh flux agent IP, thus making sure the user will be redirected to a reachable compromised machine.

Summing up, the output of the Message Prefiltering module is a list of candidate flux domains, and their related aggregated DNS information (i.e., resolved IP addresses, average TTL, etc.).

3.5 Domain Clustering

After the Message filtering module the domain names are grouped according to the similarities in their resolved sets of IPs as Botmasters usually operate malicious flux services using a(often large) number of fast-flux domain names that all point to flux agents related to the same flux network.The botmaster then registers many new domain names every day to compensate for older domain names that were identified as malicious by security operators and added to blacklists.To perform domain clustering of flux domains that are related to each other,single-linkage hierarchical clustering algorithm is used, which adopts a friends of friends clustering strategy. In order to apply the clustering algorithm to a set of domain names D ={d1, d2, . . . , dn},a measure of similarity between them is defined first.

Given two domains α and β, and their cumulative set of resolved IP addresses collected during an epoch E, respectively, $R_\alpha and R_\beta$, compute their similarity score as

$$\text{Sim}(\alpha, \beta) = \frac{|R_\alpha \cap R_\beta|}{|R_\alpha \cup R_\beta|} \cdot \frac{1}{1+e^{\gamma - min(|R_\alpha|, |R_\beta|)}} \epsilon [0, 1]$$

The first factor is the Jaccard index for sets R_α and R_β, which intuitively measures the relative overlap between the two cumulative sets of resolved IPs. The second factor is a sigmoidal weight designed to measure the confidence in estimating the Jaccard index.When the size of sets R_α and R_β is small, the Jaccard index may still have a large value if the sets exhibit a large intersection, but this value can be affected by uncertainty due to the sampling effect. Therefore, its is translated into a weight that reduces our confidence on the Jaccard index, when the resolved IP sets are small. The parameter γ is chosen a priori, according to our experience in considering a set of resolved IPs as being small or large. The value of $\gamma = 3$ workes well.

Using the similarity measure defined above, compute a similarity (or proximity) matrix P = $\{s_{ij_{i,j}}\} = 1...n$ that consists of similarities $s_{ij} = sim(d_i, d_j)$ between each pair of domains (d_i, d_j), and then apply the hierarchical clustering algorithm. At the beginning of the algorithm, each domain is considered as a cluster. Then,at each step, the two nearest clusters are merged. The algorithm stops when all the domains are included in one cluster. The obtained sequence of nested clusters is usually represented as a dendrogram, i.e., a tree-like data structure in which the leaves represent the original domains in D, and the length of the edges represent the distance between(sub)clusters.A partitioning of the set D into clusters can then be obtained by cutting the dendrogram at a certain height h.Then apply a cluster validation approach at the plateau regions.Plateau regions correspond to those steps of the algorithm where the two nearest clusters that have to be merged exhibit a quite low measure of similarity.

Summing up, the output of the Domain Clustering module is a set C = $\{C_{ii=1...l}\}$ of clusters of candidate flux domains, where domains belonging to the same clusters are related to each other, and each cluster C_i represents a candidate flux network.Then these candidate flux clusters are send to the classifier module.

3.6 Statistical Features

In order to classify a candidate flux network (i.e., a domain cluster) C, describe C in terms of a feature vector that can be processed by the statistical classifier employed in the Classifier module.

In the following, take as a reference a generic domain cluster C computed at the end of an epoch E_m, and assume that R represents the set of all distinct resolved IP addresses collected during epoch E_m that are related to the domains in C.

ϕ_1: Number of resolved IPs. Overall number of distinct resolved IP addresses in R.

ϕ_2:**Number of domains.** Total number of distinct domain names in the cluster.

ϕ_3:**Average TTL per domain.** The average TTL of the domains in the cluster.

ϕ_4:**Number of domains per network.** Number of distinct domain names observed during the previous epochs, $E_{m-1}, E_{m-2}, ...; E_{m-N}$, that share at least one resolved IP with the domains in C.

ϕ_5:**IP diversity.** Normalized entropy of the /16 network prefixes of the IPs in R, computed as follows:

$$\phi_5 = \frac{-\sum_x p(x) \cdot log_2 p(x)}{log_2(\phi_1)}$$

where the probability p(x) is given by the relative frequency of the network prefix x.

ϕ_6: **IP growth ratio.** Average number of new IP addresses discovered during E_m per each DNS query related to domains in C.

ϕ_7:**IP last growth ratio.** Average number of new IP addresses per DNS query, as discovered by analyzing the last deduplicated DNS message associated to each domain in C during E_m.

ϕ_8:**IP prefixes last growth ratio (2 features).** This feature is similar to ϕ_7, but it is based on new IP prefixes.

ϕ_9:**Novelty (3 features).** Consider the set R of resolved IP addresses observed during epoch E_m for the domains in cluster C. To compute the Novelty features, go back in time and look at the overall set of IPs pointed by domains in C during W previous epochs ($E_{m-W}, E_{m-W+1}, ...; E_{m-1}$). Let this set of IP addresses be R'. Then, compute the number of new IP addresses, i.e., IPs that belong to R but not to R', and divide this number by the number of epochs since DNS messages related to domains in C were observed for the first time. In our experiments, computed ϕ_9 for three different values of W, that is, W = 7 $(\phi_{9a}), W = 30(\phi_{9b}), andW = 180(\phi_{9c})$.

Motivation These features aim to better estimate how the set of resolved IPs (and their network prefixes) for domains in C grows in time. The larger the value of these features, the higher the likelihood that the set of IP addresses R is changing rapidly. Therefore, high values for these features provide strong support that C is related to a flux network.It is worth noting that features ϕ_5 and ϕ_8 are based on /16 IP prefixes.The rationale behind these features is that, unlike in the case of CDNs or other legitimate services, flux agents are often scattered across many networks located in many different countries, thus increasing the number of IP addresses that do not share a common /16 prefix.

3.7 Flux Classifier

At the end of each epoch E, and for each cluster Ci, measure the features described above, and employ the popular C4.5 decision-tree classifier to automatically classify a cluster Ci as either malicious flux network or legitimate/nonflux network. Reasons for using this tree:
1)They are efficient and accurate.
2)they output a set of classification rules, which allow us to asses what features are the most effective in detecting malicious flux networks.
3)C4.5 is able to automatically prune those features that are not useful or that introduce noise, rather than increasing classification accuracy.

Train the C4.5 classifier on a training data set containing a number of labeled clusters related to malicious flux services and a number of clusters related to legitimate/nonflux services. Afterward, the trained classifier is used to classify the clusters obtained at the end of each epoch E.

Chapter 4

Evaluation

4.1 Experimental setup

At first data was collected through the ISC Security Information Exchange and a labeled data set(LDS) was built. LDS is used for two purposes: 1) for estimating the accuracy of the Classifier module through tenfold cross validation, and 2) to train FluxBusters Classifier module before deployment.After that preliminary validation of the system and parameter tuning was done, and finally deployed and evaluated FluxBuster.

4.1.1 Labeled Data Set

To verify LDS a user-friendly web interface was built that allowed us to quickly sort domain clusters according to different characteristics (e.g., number of domains, number of IPs, IP diversity, etc.), verify whether (and what kind of) a website is served under a given domain, and manually label the domain clusters as either flux or nonflux. Then partially automated the labeling process by making extensive use of prior information regarding known flux domains, known malware domains, and legitimate popular domains. To reduce the number of errors (i.e., the noise) in the labeled data set, whenever a clear cut decision between flux or nonflux clusters was not possible (not even after extensive manual analysis), marked those domain clusters as unknown and excluded them from the data set. Overall, 1,337 domain clusters were labeled as flux and 5,708 as nonflux, which overall included 2,116 distinct 2LDs (59,215 FQDs) and 100,644 distinct 2LDs (113,580 FQDs), respectively.313 clusters were labeled as unknown.

4.1.2 Evaluation Data and Ground Truth

Evaluating a detection system such as FluxBuster in a live operational environment is a challenging task. The main reason is that it is not possible to obtain complete ground truth on the nature of the classified data. For example, given a domain cluster C classified by FluxBuster as flux, C may fall into three categories: 1) C includes domains and/or IPs that are known to be related to a flux network (in which case this is a true positive (TP)); 2) C does not represent a flux network, and may instead represent a CDN or other legitimate services;3) the true nature of C is unknown, that is no prior information exists on this cluster in any public (or even private) security data sources. In practice, the case when FluxBuster correctly detects a previously unknown flux network is in a way analogous to discovering a zero-day malware or exploit inthe- wild, whereby confirming the correctness of the classification usually requires manual analysis.

The main implication of the existence of these unknown cases that cannot be easily confirmed (if not with extensive forensic analysis), is that in the live evaluation there is no limit to measuring the true positive rate and false positive rate, because other quantities such as the false negative (FN) rate and true negative (TN) rate cannot be directly derived from the TP rate and FP rate,respectively. Therefore, here the true positives, false positives, false negatives, and true negatives were computed separately, using data labeled by leveraging public information about legitimate, flux, and malware-related domain names, as discussed below.

To evaluate the true positives of our classification system, we gathered two data sets of blacklisted domains: 1) a data set of known flux domains, which will referred to as the KFD data set; and 2) a data set containing known malware-related domain names, which will be referred to as the KMD data set.

The KFD data set contained 75 domain names (2LDs) classified as flux and reported by the abuse.ch website, which is well known and considered as a trusted source by many security professionals. The KMD contained 26,496 domains that were collected automatically and updated daily by harvesting public domain blacklists from 12 different reputable sources, including malwaredomains. com, malwarepatrol.com, etc. We started to collect the data included in the KFD and KMD data sets at the end of August 2010, and since then we have kept them up-todate with daily updates. The rationale behind using the KMD is that although this data set contains many domain names that are not related to flux networks, if a malware domain m passes the prefiltering stage (thus becoming a candidate flux domain) and also FluxBuster flags a domain cluster that includes m as having the characteristics of a flux network, it is extremely likely that the cluster is indeed a flux network (we manually confirmed a large number of these cases).

To evaluate the false positives, we collected three data sets of whitelisted domains: 1) a data set containing the top 100,000 domain names according to Alexa.com, which we refer as the ATD data set; 2) a data set containing 304,248 distinct second-level domains (321,411 fully qualified domain names) drawn at random from the Yahoo! DMOZ project (which lists manually verified websites), referred to as YDD data set; and 3) a manually compiled CDN data set containing a list of 31 second-level domains related to both well known and less well-known legitimate CDNs. The ATD data set actually contained 57,910 domains that are consistently top, i.e., those domains that remained within the top 100k ranking for almost one entire year. The assumption is that the consistently top domains are legitimate, since malicious domains typically have a short life span (this is particularly true for the popular ones that attract the attention of the security community). In practice, if FluxBuster classifies a domain cluster C as nonflux while the domains in the cluster belong to KFD or KMD, consider this to be a false negative. It is worth noting that this approach may overestimate FluxBusters false negatives, because most of the domains in KMD are not flux domains. However, since FluxBusters Classifier module only receives candidate flux networks from the previous modules (see Section 3.1), assume that if a malware domain belongs to a candidate flux network there is a high probability that the domain is actually flux. On the other hand, to compute the true negatives leveraged the ATD, YDD, and CDN data sets. In this case, things are more straightforward: domains from these three data sets are highly likely legitimate, and therefore if FluxBuster classifies a domain cluster containing any of these domains as nonflux, so the classifier was indeed correct.

4.2 Experimental Results

4.2.1 Cross-Validation

Using the labeled data set LDS,a tenfold cross validation was performed to estimate the accuracy of the Classifier module. The decision tree classifier was able to achieve 99.3 percent detection rate at 0.15 percent false positive rate, with an area under the ROC curve (AUC) of 0.994. These results confirm that FluxBuster can detect malicious flux networks with high accuracy.

4.2.2 Live Evaluation

Along with the tenfold cross validation on the labeled data set LDS, detection system was evaluated in a real-world deployment scenario. To this end, first trained FluxBuster on the entire LDS data set, and then deployed it at ISC/ SIE. Overall, in a period of about five months of operational deployment, FluxBuster classified 4,084 domain clusters as flux and 3,633 domain clusters as nonflux, which included a total of 1,743 2LDs (63,442 FQDs) and 227,667 2LDs (264,550 FQDs), respectively. These results were obtained for a threshold k = 30.

True positives

1)Known flux. The number of domains classified as flux by FluxBuster was computed that were also in the KFD data set. It was found that FluxBuster correctly classified 24 out of 75 KFD domains (2LDs) as flux. The remaining 51 domains were not classified by FluxBuster, simply because they were not visible from the SIE data feed.Then computed how many new flux domains were able to detect, using the KFD domains as a seed. In practice, starting from KFD, first they found the setFC of domain clusters classified by FluxBuster as flux that contained at least a domain d ϵ KFD. Then, computed how many of the domains in these FC clusters were missing from KFD. Using this approach,they were able to find a total of 525 new flux domains that belong to a malicious flux network also pointed by domains in KFD. Furthermore, it was found that of these 525 new flux domains, 79 of them were known malware domains belonging to the KMD data set. Following a natural guilty by association rule, the remaining 446 can also be regarded as new malicious flux domains.

2) Known malware.It was found that a total of 179 domains (2LDs) in the KMD data set were classified as flux by FluxBuster.

False Positives

First checked how many domains classified as flux by FluxBuster were also present in the ATD data set. It was found that only two fully qualified domains out of the 57,910 2LDs in ATD were classified as flux.Afterward, checked how many domains classified as flux were part of the CDN data set. Over the entire evaluation period (almost five months), it was found that only 35 fully qualified domains under four distinct 2LDs caused false positives. These four 2LDs were related to small CDNs.The flux domains against the YDD data set, and found no false positives in this case.

True Negatives

Overall, FluxBuster classified a total of 3,633 domain clusters as nonflux, which included a total of 227,667 2LDs (264,550 FQDs).It was noticed that many clusters contained large numbers of legitimate domains that shared the same set of IPs because they all shared the same physical Web server. For example, it was found large clusters of personal websites that appeared to rely on virtual hosting services.

False Negatives

To measure the false negatives,checked how many of the domains belonging to the KFD data set were consistently classified by FluxBuster as nonflux.It was found that only one such domain belonged to KFD: discountpharmacyhealth.net. This false negative was due to the fact that discountpharmacyhealth. net resolved into a relatively low number of distinct IP addresses (less than 60 overall), had a low value of the novelty features, and a relatively low IP diversity. Similarly,computed how many of the domains consistently classified as nonflux were in the KMD data set.It was found 30 such domains (out of the 26,496 domains in the data set).

4.2.3 Early Detection Results

Overall, it was found that FluxBuster detected nine out of 24 confirmed flux domains earlier than when they appeared in the KFD data set (the remaining 15 domains were detected later than they appeared in KFD), and 125 out of 179 confirmed malware domains earlier than when they appeared in KMD.To gain a better view of how earlier FluxBuster was able to detect these malicious domains, it was proceeded as follows: let d be a flux domain, t_{FB} be the day when d was detected by FluxBuster, t_{KFD} be the day when d appeared in KFD, and t_{KMD} be the day when d appeared in KMD. Fig. 4 shows how many domains were able to detect for each value of $\delta_{KFD} = t_{KFD} - t_{FB}$, while Fig. 5 shows how many domains were able to detect for each value of $\delta_{KMD} = (t_{KMD} - t_{FB})$.

As the graphs show,FluxBuster is often able to detect malicious flux domains days or even weeks before they are detected by well-known third-party security services. In addition,domain names were isolated related to the infamous Zeus botnet from the KMD data set. It was found that FluxBuster detected 13 out of 21 Zeus domains earlier then when they appeared in KMD. Fig. 6 shows the distribution of early detection days for these domains.

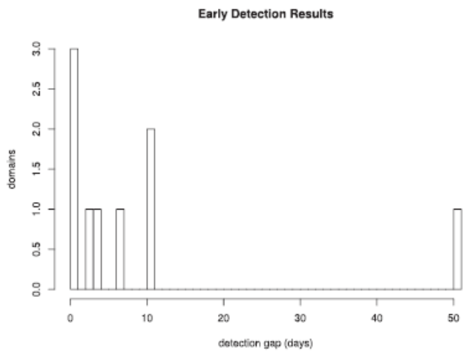


Fig. 4. Early detection of **KFD** domains.

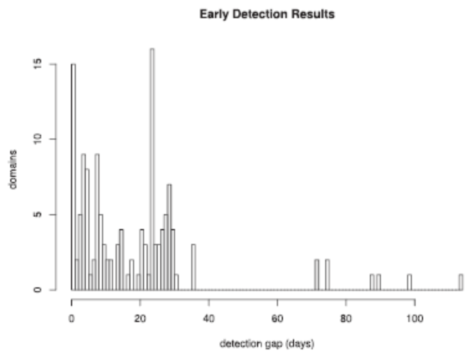


Fig. 5. Early detection of **KMD** domains.

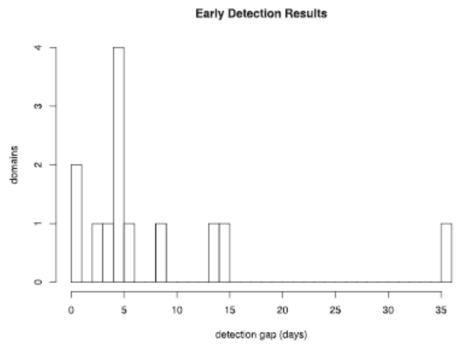


Fig. 6. Early detection of Zeus botnet domains.

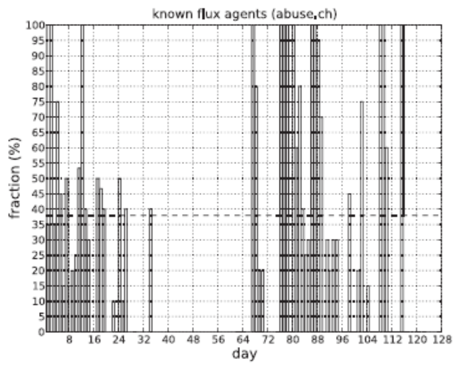


Fig. 7. Fraction of flux agents among those detected by FluxBuster

4.2.4 IP-Based Analysis

Using the online DNSBL lookup service7 provided by abuse.ch,the fraction of flux agents were measured (i.e., IPs) enumerated by FluxBuster that were known or not to abuse.ch. In particular, for each epoch, at the end of FluxBusters classification process, randomly sample 10 flux agent IPs from the clusters labeled as flux, and query abuse.chs DNSBL on this IPs. It is worth noting that only consider flux clusters that contain at least one domain name that is also present in KFD (i.e., the data set of known top flux domains reported by abuse.ch itself). Fig. 7 shows, per day, the fraction of IPs (i.e., flux agents) enumerated by FluxBuster that were listed as malicious by abuse.ch. It was found that, in average, about 62 percent of the tested IPs were actually unknown to abuse.ch. This suggests again that FluxBuster can offer a valuable complementary view of flux networks in terms of flux domain names and their flux agents.To this end,first considered (per each day) the set P of IP addresses (i.e., flux agents) contained in flux clusters that included at least one domain name belonging to the KFD or KMD data sets. Then,retrieved (per each day) the set of all the domain clusters C ={ C1, C2, . . . , Cn } that were classified as flux by FluxBuster and that contained at least one domain p ϵ P. Finally,counted the number of distinct single domain names contained in the flux clusters in C that were not present in KFD or KMD. It was found 1,030 such domains, and was verified that they were mostly related to pharmacy scams and porn-related websites, and that none of these domains appeared in ATD, YDD, and CDN. In practice, through this process it was possible to confirm the fact that FluxBuster was able to discover 1,030 previously unknown malicious flux domain names.

4.2.5 Flux Websites

It was found that the vast majority of flux domains detected by FluxBuster resolve to flux agents that successfully respond to HTTP requests (on default port 80) and return malicious content/scam pages. Such domains represent a threat to web users. To reduce these threats, FluxBuster could provide an interface similar to the Google Safebrowsing API,which can be queried by a browser plugin to inform users that they are about to visit a website under a flux domain.

To have a sense of the potential benefits of using FluxBuster under this browser-plugin setting,counted the number of effective 2LDs that classified as flux were hosting web content.Fig. 8 shows the results obtained over almost one month of data

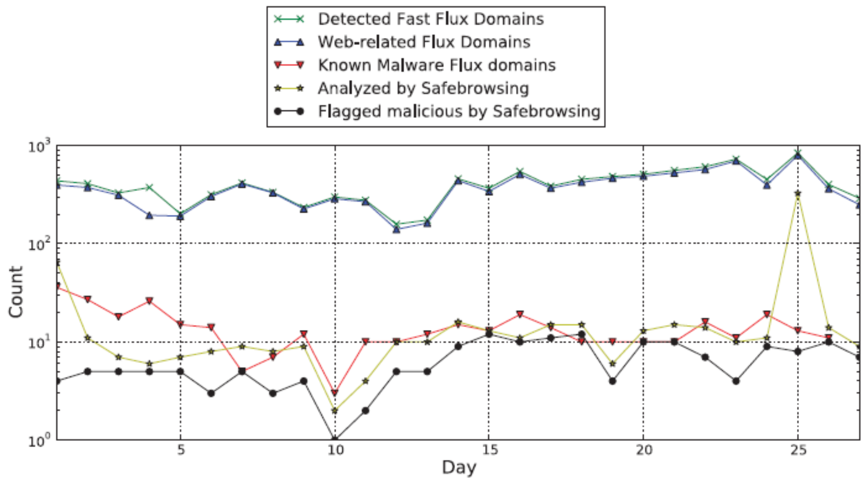


Fig. 8. Analysis of flux domains that host web content.

The Web-related Flux Domains represent domain names for which some web content was successfully downloaded. To perform this measurement, for each fast flux domain name and issued a number of HTTP GET requests on the root URL /, and on other related URLs obtained by searching for the domain name on search engines.9 The Known Malware Flux Domains represent the number of flux domains that also appear in either KFD or KMD. To compute the Analyzed by Safebrowsing domains that were queried each flux domain against the Google Safebrowsing service, and the number of domains were counted that Google Safebrowsing reported as previously tested. On the other hand, the Flagged Malicious by Safebrowsing domains are the ones that have been tested by Google Safebrowsing, and have been labeled as malicious.

This analysis aimed to show that FluxBuster can complement other public web security services by providing additional useful information to users who prefer to avoid visiting any kind of potentially malicious websites (including rogue pharmacies, suspicious porn-related sites, and other scam websites), while browsing the Web.

Chapter 5

Conclusion

The above evaluation showed that FluxBuster is capable of accurately detecting previously unknown flux networks days or even weeks in advanced before they appear in public blacklists.FluxBuster also detects flux domains that may remain unknown to other third-party systems, thus providing valuable information to the security community that can be used to block a variety of malicious content.

5.1 Future Scope

The evaluation results show that FluxBuster is able to detect flux networks in-the-wild, and can often do so several days or even weeks earlier than other state-of-the-art detection systems. This does not mean, though, that FluxBuster should replace other flux and malware detection systems. Instead,FluxBuster was designed to complement other detection systems, and, in particular, to compensate for some of the limitations of flux detection systems based on active probing.

FluxBuster has its own limitations. For example, for a flux domain to be detected, FluxBuster needs to observe at a minimum one DNS message related to that domains. Because FluxBusters observations depend on users behavior (e.g., clicking on domain names), if a flux domain is never queried by any of the users within the monitored networks, FluxBuster will not be able to detect it. Also, enough IP addresses need to be collected during a given epoch (e.g., one day) to perform the classification of a domain cluster. This may cause some false negatives.However, FluxBuster aims to detect flux domains, and more in general flux networks, that are active and successfully attract victim users in at least one epoch.Currently, FluxBuster can process one day worth of DNS traffic collected from hundreds of distributed ISC/SIE sensors in about eight hours. As a future work, we can study how to shorten FluxBuster's response time.

References

[1] Roberto Perdisci, Igino Corona, and Giorgio Giacinto. Early Detection of Malicious Flux Networks via Large-Scale Passive DNS Traffic Analysis. In IEEE Transactions on Dependable and Secure Computing, VOL. 9, NO. 5, September/October 2012.

[2] M. Knysz, X. Hu, and K.G. Shin.Good Guys vs. Bot Guise: Mimicry Attacks against Fast-Flux Detection Systems.In Proc. IEEE INFOCOM, 2011.

[3] T. Holz, C. Gorecki, K. Rieck, and F. Freiling.Measuring and Detecting Fast-Flux Service Networks. In Proc. Network and Distributed System Security Symp. (NDSS 08), 2008.

[4] J. Nazario and T. Holz.As the Net Churns: Fast-Flux Botnet Observations.In Proc. Third Intl Conf.In Malicious and Unwanted Software (MALWARE 08), 2008.