Automatic detection of DNS manipulations

Automatic Detection of DNS Manipulations

Martino Trevisan, Idilio Drago, Marco Mellia, Maurizio M. Munafò
Politecnico di Torino, Italy
{firstname.lastname}@polito.it

***Abstract*—The DNS is a fundamental service that has been repeatedly attacked and abused. DNS manipulation is a prominent case: Recursive DNS resolvers are deployed to explicitly return manipulated answers to users' queries. While DNS manipulation is used for legitimate reasons too (e.g., parental control), *rogue DNS resolvers* support malicious activities, such as malware and viruses, exposing users to phishing and content injection.**

We introduce REMeDy, a system that assists operators to identify the use of rogue DNS resolvers in their networks. REMeDy is a completely automatic and parameter-free system that evaluates the consistency of responses across the resolvers active in the network. It operates by passively analyzing DNS traffic and, as such, requires no active probing of third-party servers. REMeDy is able to detect resolvers that manipulate answers, including resolvers that affect unpopular domains.

We validate REMeDy using large-scale DNS traces collected in ISP networks where more than 100 resolvers are regularly used by customers. REMeDy automatically identifies regular resolvers, and pinpoint manipulated responses. Among those, we identify both legitimate services that offer additional protection to clients, and resolvers under the control of malwares that steer traffic with likely malicious goals.

I. Introduction

The DNS is fundamental to the Internet, since it translates Fully Qualified Domain Names (FQDNs) into IP addresses. The DNS is a large-scale system composed of millions of servers that interact to resolve users' queries. Operators usually deploy *recursive* DNS resolvers to handle the resolution of queries close to their users. It is however common to find users that intentionally opt for open recursive resolvers (e.g., Google's DNS or OpenDNS) or, perhaps unintentionally, rely on arbitrary resolvers. The latter may happen, for instance, when software installed in users' machines change DNS configurations automatically, e.g., some parental control solutions that block access to non-authorized or unsafe domains.

Recursive DNS resolvers are expected to answer users' queries by reaching the *authoritative* resolver for the FQDN. However the DNS has been repeatedly attacked and abused. Among the millions of open DNS resolvers in the Internet, it is strikingly common to find resolvers that *manipulate* responses [4], [9]. In other words, some recursive resolvers sometimes answer queries with responses that diverge from what is returned by the authoritative ones. Such manipulations may occur due to legitimate reasons (e.g., parental control), but a variety of malware and viruses change clients' DNS setup too, e.g., to force users into phishing websites or to perform content injection. This latter case is illustrated in Fig. 1, where an infected client (yellow) contacts a malicious DNS resolver and queries for a sensitive FQDN (e.g., *mybank.com*), but receives the IP address of a fake web server as answer. Whereas

This research has been funded by the "SmartData@PoliTO" center.

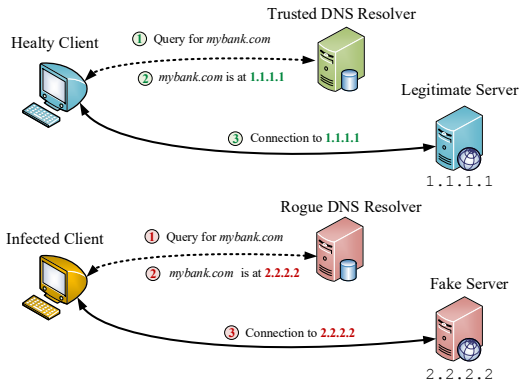

Fig. 1: DNS manipulation through forged DNS responses.

solutions such as DNSSEC provide the means for clients to check authenticity and integrity of response and, thus, detect DNS manipulations, they are still barely deployed [6], letting users exposed to such malicious *rogue DNS resolvers*.

It is thus not a surprise that several studies have focused on DNS anomalies. Some works propose generic methods to identify anomalies in the DNS [1], [2], [7], such as DNS cache poisoning. Regarding DNS manipulations, seminal works [3], [12] use active measurements and manual heuristics to profile open resolvers. Similar methodology is employed by authors of [13], who also download pages from web servers hosted in the IP addresses returned by open resolvers, searching for anomalies and signs of manipulations. More recently, authors of [10], [4] have revisited the problem proposing new active methodologies to detect manipulations. They have scanned the complete IPv4 Internet and found millions of open resolvers that manipulate responses. Similar steps are performed by a system called Iris [9] that aims at identifying the use of DNS manipulations for implementing censorship in the Internet.

Previous works however require active probing of external servers. We here face the question on whether and how DNS manipulations can be detected directly from the DNS traffic. This would allow network operators to identify clients exposed to manipulations without the burdens of periodically probing external hosts for suspicious DNS resolutions.

We thus introduce REMeDy, REsolver Manipulation Detector, a system to help operators identify the use of rogue DNS resolvers in their networks. The identification of rogue resolvers from passive traffic observation comes with many challenges. First, DNS load balancing techniques are commonly used by Content Delivery Networks (CDNs) and cloud systems, so that answers may vary widely, e.g., changing over time and based on the geographical position of the client

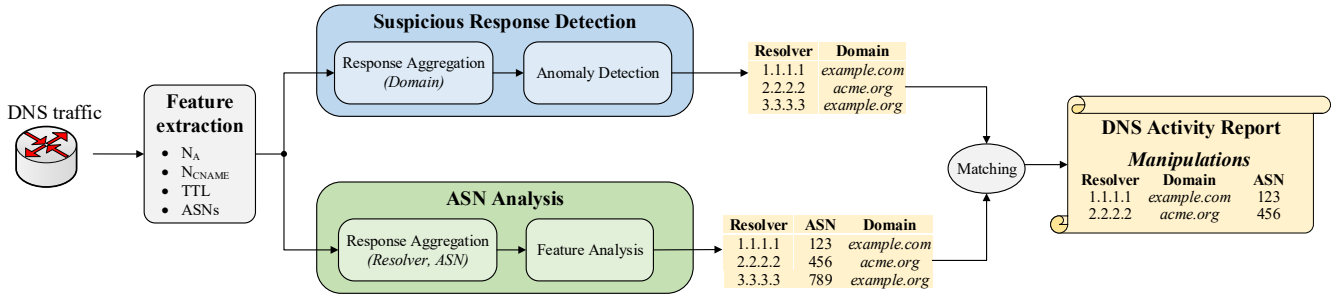

Fig. 2: REMeDy architecture and building blocks.

and DNS resolver. Second, malicious resolvers may return perfectly legitimate answers to the large majority of queries, while modifying only a small subset of queries, e.g., for very specific FQDNs involved on click fraud campaigns.

REMeDy is completely automatic and parameter-free. It evaluates the consistency of responses across all resolvers by extracting features directly from the DNS traffic. These features include the number and type of records on DNS responses, typical values of Time To Live (TTL) fields, and the Autonomous System (AS) of returned IP addresses. We rely on statistical methodologies to pinpoint deviation from the legitimate answers, which in turn are automatically identified from the traffic.

We validate REMeDy by facing a case study. We let it run on traffic traces collected from operational networks where the most used resolvers have been manually profiled and the manipulating resolvers have been manually identified. We show that REMeDy is able to detect the manipulating resolvers, including some legitimate and well known services (e.g., providing parental control, or antivirus filtering services). Finally, REMeDy is able to pinpoint malicious resolvers that only sporadically manipulate answers to force users to go through fake servers likely under the control of attackers.

II. REMeDy detection algorithm

REMeDy aims at finding manipulated DNS responses, including those provided by rogue resolvers that divert traffic to specific malicious servers. It processes batches of DNS traffic from a (potentially large) network and outputs detailed insights about DNS anomalies. REMeDy is parameter-free and completely unsupervised as it does not require any ground truth for training. Its architecture is depicted in Fig. 2.

The system operates in four stages. First, it extracts features from live DNS traffic passively observed in the network. Second, it aggregates responses in a per-$Domain$ fashion, and computes statistics about the resolutions. Statistical-based anomaly detection unveils uncommon DNS resolutions, which are marked as *suspicious*. Third, to filter out false positives (e.g., anomalous responses due to CDNs, load balancing and unpopular resolvers), REMeDy aggregates responses per-$(Resolver, AS)$, thus evaluating properties of destinations where users are directed when resolving domains. Finally, resolutions tagged as anomalous by both stages are used to determine the final list of manipulators, and results are presented along with statistics that explain the source of the anomalies.

We next detail each of these four stages. Table I summarizes the terminology used in the following sections.

TABLE I: Main terminology used throughout the section.

R	The IP address of a recursive resolver
D	The queried domain name (i.e., a FQDN)
f	A feature extracted from DNS response packets
$Freq(D, f)$	Frequency distribution of feature f for responses to queries for domain D regardless of resolver
$Freq(R, ASN, f)$	Frequency distribution of feature f for the resolver R when resolving to ASN

A. Feature extraction

REMeDy is given as input DNS responses observed in the monitored network, e.g., traffic captured using passive monitoring tools. The system processes traffic in batches, i.e., it groups all DNS responses observed in a period of time (e.g., 1 day), and extracts features to identify DNS manipulations.

DNS responses are parsed to inspect the DNS Resource Records (RRs); a RR is the basic information element of the DNS protocol. From each DNS response, REMeDy extracts R and D, respectively the IP address of the resolver and the queried domain. Then, the actual features are retrieved. REMeDy considers 4 features:

- N_A, the number of `A` (IPv4) and `AAAA` (IPv6) records in the response;
- N_{CNAME}, the number of `CNAME` records used to specify domain aliases in the response;
- TTL of `A` or `AAAA` records. When several records are present, we take the maximum TTL, although in practice all records usually have the same TTL within a packet;
- $ASNs$, the list of Autonomous System Numbers (ASNs) of server IP addresses in `A` and `AAAA` records. Resolvers often return many `A` or `AAAA` records, e.g., when the queried domain is hosted on multiple servers.

All features are computed directly from the RRs, except $ASNs$. We further process returned server IP addresses to improve the detection of manipulations. Popular web services rely on big infrastructures to cope with the workload of a large number of users. Some services take advantage of cloud providers and CDNs that deploy thousands of servers hosted in many networks. As a consequence, the several queries for a single domain are often resolved to diverse IP addresses located in different data-centers, networks and possibly

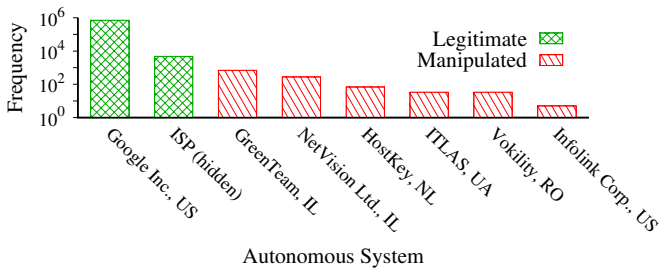

Fig. 3: $Freq(www.google\text{-}analytics.com, ASNs)$: frequency distribution of $ASNs$ for a domain of Google Analytics.

ASes [11]. This behavior makes it virtually impossible to detect anomalous resolutions based on IP addresses.

Thus, we replace IP addresses by ASNs to reduce the variability in the list of returned IP addresses. The mapping from IP addresses to ASNs can be done online through services like Whois and Team Cymru. Alternatively, platforms like BGPStream [8] can be leveraged to create a local database of ASNs.[1]

B. *Suspicious response* detection

The detection of suspicious responses is performed in two steps. First, REMeDy consolidates statistics across all responses related to each domain D. Then, the distributions of values for each feature are evaluated and the pairs of resolver and domain, i.e., (R,D), responsible for the outliers are included in a list of *suspicious* resolutions.

1) Frequency distributions: The first step of REMeDy is to consolidate statistics for each queried domain. For this, it aggregates features for all responses matching each domain D. The intuition is that the majority of resolvers will provide answers with similar features, and only those that diverge from the majority are worth evaluating for possible manipulations.

REMeDy collects the responses given D and computes the frequency distribution $Freq(D, f)$ for the observed values of each feature f. For instance, considering the $ASNs$ in all responses matching D, REMeDy computes a vector containing how often each ASN is present in responses to queries for D. Note that these statistics are evaluated taking into account all responses, regardless the resolver.

2) Anomaly detection: REMeDy evaluates $Freq(D, f)$, for all D, and outputs the list of suspicious resolvers. One possible approach to detect suspicious resolvers could be (i) to calculate the frequency distributions for D and f considering each resolver R, and (ii) to compare these conditional distributions against $Freq(D, f)$. Unfortunately, a direct comparison is not effective since the number of samples can be very low for unpopular resolvers. As such, unpopular resolvers would often be marked as suspicious.

REMeDy focuses on outliers of the distributions $Freq(D, f)$. The intuition is illustrated in Fig. 3, which depicts $Freq(D, f)$ for D = *www.google-analytics.com* and $f = ASNn$. This figure is built with a sample of the DNS traffic used for validating the performance of the system.

[1]We tested other grouping alternatives, e.g., by /24 subnet, which result in rather noisy groups. When the ASN information is missing we use /16 subnet.

TABLE II: TTL_{MAX} for *www.google-analytics.com*.

Resolver	Maximum TTL [sec]
ISP DNS 1	300
ISP DNS 2	300
Google DNS	300
OpenDNS 1	300
AdWare	38400

Note the logarithmic y-axis. For the vast majority of queries, resolvers answer with IP addresses belonging to the ASNs of Google or the Internet Service Provider (ISP) in which the data has been collected. The latter happens because of CDN nodes hosted in the ISP network. A long tail of unpopular $ASNs$ is also visible. Manual inspection reveals that those unpopular $ASNs$ are in manipulated responses connected to malwares that divert traffic to fake web servers.

REMeDy needs to identify these groups automatically for all features f, and without any training. REMeDy takes two distinct approaches depending on the feature.

a) Categorical features: $ASNs$ is clearly a categorical feature, as only the equivalence relation can be defined. Although numerical, N_A and N_{CNAME} typically assume values smaller than 10, thus suggesting considering them categorical features too, rather than numerical.

To discover anomalies from categorical features REMeDy relies on classical techniques to find outliers. We build on the well known *box plot rule* [5], which leverages the lower (Q_1) and the upper quartiles (Q_3) of a distribution to define the *Inter Quartile Range* (IQR) as $IQR = Q_3 - Q_1$. Then, all instances lying below $Q_1 - 1.5 \cdot IQR$ or above $Q_3 + 1.5 \cdot IQR$ are considered outliers.[2]

We apply the technique to values of $Freq(D, f)$ to find outliers given each domain D and feature f. Since the distributions can only take positive values, we consider outliers all those values of features lying above $Q_3 + 1.5 \cdot IQR$ for the respective $Freq(D, f)$. Then, REMeDy considers as suspicious *any* pair (R,D) behind a DNS response with a categorical feature that falls in the range of outliers.

b) The TTL feature: TTL is a numerical feature, in particular a number expressed in seconds. It thus allows central tendency and dispersion measures. However, given the machine-generated nature of the feature, some ingenuity helps to improve the quality of the anomaly detector.

Each DNS response carries a TTL computed from the one originally set by the authoritative name server. When asked for a domain, a recursive resolver queries the authoritative server and, once the response is obtained, caches it until the TTL expires. A subsequent request for the domain will be quickly answered, as it is locally resolved without involving the authoritative server. This second response carries exactly the same RRs as the original one, except for the TTL that is decreased by the time elapsed between the responses. The resolver invalidates the local cache when the TTL expires and recontacts the authoritative server, thus renewing the TTL. Because of this caching mechanism, TTL values smaller than what is set by the authoritative resolvers are completely legitimate and common.

[2]We tested other approaches, all leading to worse final results.

TABLE III: $Freq(R, ASN, f)$ for a R that is known to be a rogue resolver. The last line corresponds to forged resolutions.

ASN	AS Name	N_A	N_{CNAME}	TTL	Resolved Domains
15169	Google Inc., US	1,2,3,4,6,8,10...	0,1,2	$[0-30]$	www.google.com, www.youtube.com, gmail.com...
32934	Facebook, Inc., US	1	0,1,2	$[0-3600]$	www.facebook.com, connect.facebook.net, static.xx.fbcdn.net...
20940	AKAMAI-ASN1, US	1,2	0,1,2,3,4,5	$[0-43200]$	www.microsoft.com, i.alicdn.com, cdncache-a.akamaihd.net...
1680	NetVision Ltd., Israel	1	0	300	www.google-analytics.com, gstatic.com, static.chartbeat.com...

REMeDy uses this domain knowledge to estimate the original TTL set by the authoritative server for each domain D. In particular, we define $TTL_{MAX}(R, D)$ to be the maximum TTL observed for the resolver R and domain D. We then can compute $Freq(D, f = \text{“}TTL_{MAX}\text{”})$, as the frequency distribution of TTL_{MAX} across all resolvers for the domain D. The intuition is illustrated in Table II, which shows the $TTL_{MAX}(R, D)$ for $D =$ *www.google-analytics.com* and five resolvers. The same sample dataset of Fig. 3 is used. For most resolvers, we observe $TTL_{MAX}(R, D) = 300$, suggesting that this is the value set by the authoritative server. The (abnormally) high value set by last resolver is suspicious.

Once $Freq(D, f = \text{“}TTL_{MAX}\text{”})$ is calculated, the detection of outliers is performed following again the box plot rule. REMeDy then adds to the list of suspicious resolutions any pair (R,D) with TTL in the range marked as outlier for the TTL_{MAX} of domain D.

C. ASN analysis

The set of (R,D) that are marked as suspicious includes manipulations, but also false positives. This is expected since in the previous stage REMeDy considers suspicious all resolutions diverging from the majority. However, some users may opt to use resolvers located in networks that are physically far away from the place where they are connected. Resolvers in different places may behave differently. For instance, the remote resolver may provide optimized responses based on its location, e.g., redirecting users to CDN nodes that are close to it. Such resolutions are put in the suspicious list too.

Thus, REMeDy performs a second round of anomaly detection to polish the suspicious list. While the previous stage focuses on how each domain is resolved by the several resolvers, this second stage focuses on finding resolutions that are redirected to suspicious autonomous systems.

The intuition is simple. Each resolver answers queries related to a variety of web services that are hosted in different autonomous systems. If an AS hosts multiples popular domains, the features characterizing responses pointing to the AS are expected to present variability, e.g., the TTL should vary from response to response. Resolvers that manipulate responses alter this behavior. For example, malware returns IP addresses belonging to the ASes hosting attacker's servers for various domains. Hence, features of such responses do not follow general patterns, i.e., present little variability.

REMeDy implements a simple rule proven sufficient to filter out false positives from the list of suspicious (R,D) pairs.

From original features, REMeDy calculates statistics in a per-$(Resolver, ASN)$ level, thus analyzing together all resolutions of a resolver R that point to each ASN. REMeDy calculates $Freq(R, ASN, f)$, i.e., the frequency distribution for the feature f, considering responses in which the resolver R has returned the given ASN. Then, REMeDy isolates the cases in which $Freq(R, ASN, f)$ has no variability for at least two features – i.e., two features among N_A, N_{CNAME} and TTL present *always* the same value, given (R, ASN).

Table III illustrates the mechanism with an example related to a rogue resolver. This particular resolver manipulates responses only for a subset of domains. When one of such domains is queried, the resolver redirects users to the ASN 1680. Otherwise, legitimate ASes are returned. The table shows values of N_A, N_{CNAME} and TTL for the $ASNs$ returned by the resolver, as well as some of the queried domains. We see that when the resolver is providing legitimate answers, features exhibit variability (lines 1,2,3). On the contrary, forged responses have fixed feature values (line 4).

The analysis described above comes with an extra caveat. $Freq(R, ASN, f)$ will naturally present little or no variability if only a few samples are present, as well as if the given ASN hosts only a couple of domains that are seldom resolved. In other words, REMeDy needs to ignore $Freq(R, ASN, f)$ if a low number of resolutions points to ASN, or if only few domains are present in responses leading to ASN.

$Freq(R, ASN, f)$ is thus considered valid only if two conditions are satisfied. Given (R, ASN), (i) the number of domains resolved to the ASN must be large; (ii) the average number of resolutions per domain leading to ASN must be large. Clearly, thresholds are needed to define *large* in both cases. Since our goal is to make REMeDy completely parameter-free, we adopt a simple rule-of-thumb to determine the thresholds. We consider $Freq(R, ASN, f)$ valid if those quantities are above the median values computed from all resolutions that are *not* in the suspicious list of the previous stage. Considering metrics other than median does not significantly change the final result.

Pairs (R, D) that are in resolutions leading to valid samples of $Freq(R, ASN, f)$ filter out the suspicious list. In other words, only pairs of resolver and domain (R, D) passing both anomaly detection stages are declared manipulations.

D. DNS activity report

Resolvers that pass both anomaly detection stages are consider manipulators. REMeDy creates a per-resolver report and present it to the analyst. The report provides a summary of the global DNS activity (e.g., the number of queries) and details about the features leading to the tagging of particular resolvers (e.g., domains presenting anomalous features).

The latter allows the analyst to spot the websites manipulated by the resolver. We will show next that by evaluating the manipulated features, one can already gain some knowledge about the type of manipulation performed by the resolver.

TABLE IV: DNS traces from operational networks.

Name	Duration	Log Size	Queries	Clients	Resolvers
ISP1	7 Days	173 GB	273 M	7,500	690
ISP2	7 Days	51 GB	172 M	5,000	145
Campus	7 Days	186 GB	135 M	10,000	199

III. Validation

A. Datasets

We rely on 3 datasets coming from operational networks. All datasets have been captured in 2017 from vantage points that observe all DSN traffic generated by a population of users.

We use Tstat[3], a passive monitor that exports flow level statistics. For DNS traffic, Tstat can also export packet-level information, including the Transaction Identifier, Response Code and flags of DNS responses. Moreover, Tstat exports details about Resource Records (RR) in each packet: Query, (eventual) Answer, Type of Record, Class, Time-To-Live, etc.

Table IV summarizes the datasets. ISP1 and ISP2 come from an European ISP. Respectively 7,500 and 5,000 ADSL and Fiber-To-The-Home customers are monitored. The third dataset, Campus, comes from a University Campus hosting more than 10,000 people.

Captures have been performed during one full week. Users in these networks have contacted hundreds of DNS resolvers in the week. More than 500 million queries to the DNS have been performed during the captures.

B. REMeDy performance

Obtaining the ground truth to validate REMeDy is not trivial. No service there exists providing classification of open resolvers, nor there are exhaustive lists of the rogue ones. Because of that, we validate REMeDy directly by deploying it in production and manually evaluating its outcomes.

We make use of ISP1 trace, and manually check all resolvers generating more than 1,000 response packets. Among the 690 resolvers observed during the week of analysis, we have investigated 111 resolvers manually. We have collected information about them from various sources: by searching the Web for information, by performing active measurements to verify the content of resolutions and by investigating the websites of companies hosting the resolvers.

We have found that 14 resolvers are deployed by the monitored ISPs, 43 are open resolvers serving different purposes, and 33 resolvers are related to some kind of malware. Finally, 21 resolvers are used by antivirus clients for DNS tunneling.

Anomalies reported by REMeDy are summarized in Table V. REMeDy correctly finds no anomalies among the ISP resolvers and DNS tunnels.

REMeDy finds 7 anomalies among open resolvers. We have tracked these cases to content filtering. Security suites like Comodo or Norton deploy resolvers that implement parental control or prevent users from contacting malicious domains. Filtering is performed by steering the traffic of particular domains towards a controlled server. As a result, this behavior is marked as anomalous by REMeDy. Even if not related to malicious activity, these *are* cases of DNS manipulations.

[3]http://tstat.tlc.polito.it

TABLE V: Number of detected manipulations in ISP1.

Type	#Resolvers	#Manipulating
ISP resolver	14	0
Open resolver	43	7
Malware	33	19
Tunnels	21	0

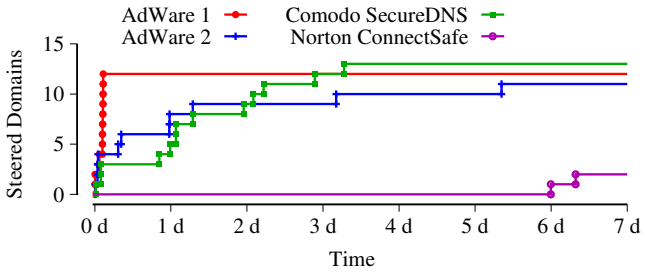

Fig. 4: Number of manipulated domains as a function of time in ISP1 trace.

Among 33 resolvers related to malware, the system finds anomalies in 19. REMeDy does not find anomalies in 14 malware resolvers. Manual inspection points to the following reasons: (i) the number of observations for (R, ASN) is still below the thresholds (8 cases), (ii) no domain apparently manipulated – i.e., the malware was likely inactive during the captures (1 case), (iii) resolvers use different mechanisms to hijack the traffic, without manipulating legitimate requests, e.g., they hijack only NXDOMAIN responses (5 cases). The last two cases are out of scope for REMeDy, whereas the first would eventually be detected if longer monitoring is performed. We conclude that, when enough traffic is observed, REMeDy is able to identify forged DNS resolutions.

Finally, we evaluate the time needed by REMeDy to detect manipulations. Fig. 4 focuses on 4 resolvers that are active during the whole duration of the capture. We plot the number of domains that the resolver has manipulated as a function of the capture time. The figure shows that some resolvers manipulate popular domains and, thus, are detected quickly. AdWare 1 manipulates 12 domains, and it has already been identified in the first hours of the capture. Other resolvers focus on less popular domains. Norton's resolver only shows signs of manipulations after six days. This happens because, being a completely passive system, the detection of DNS manipulations depends of the users' activity.

C. Which domains are manipulated?

We now evaluate which domains are typically manipulated. We report the results obtained when running REMeDy on the three datasets. REMeDy identifies 37 manipulating resolvers in total, some of them present in more than one dataset.

Seven resolvers perform legitimate manipulations related to content filtering as explained above. The remaining 30 resolvers are related to AdWare malwares. They manipulate responses related to advertising and tracking services in order to inject content (e.g., ads) while victims browse the web.

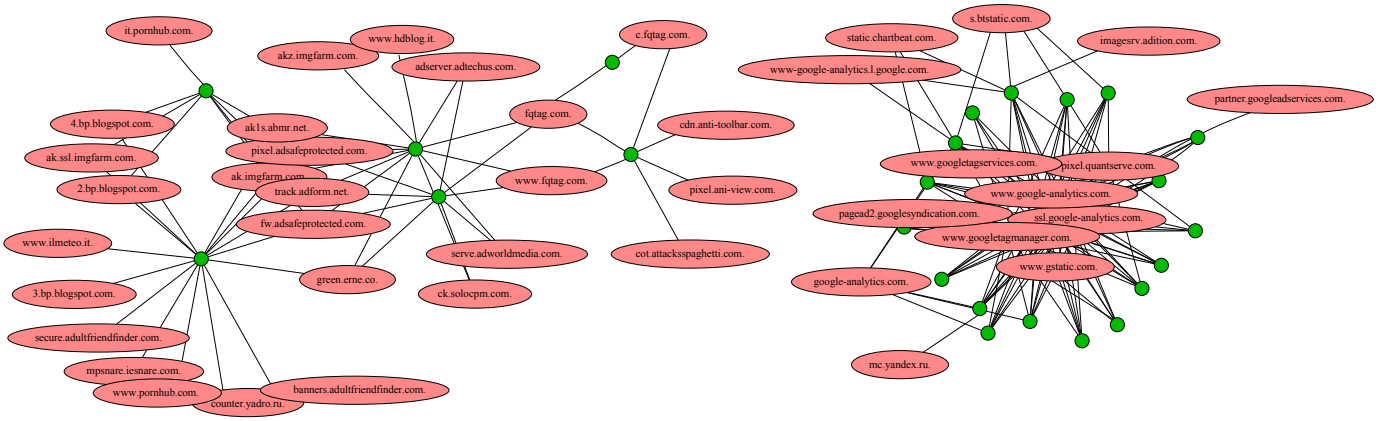

Fig. 5: Manipulated resolutions found in ISP1 dataset. Resolvers are green circles, domains are red.

In particular, we find a massive infection caused by DNS-Unlocker AdWare involving more than 40 clients and 18 rogue resolvers.[4]

In Fig. 5 we represent the manipulations discovered in ISP1 as a graph, connecting resolvers to the domains they manipulate. Two clusters are evident: AdWare resolvers manipulate advertising domains (right-hand side), while (legitimate) security suites manipulate traffic related to malware and adult websites (left-hand side).

Such visualization allows analysts to separate benign manipulations from the malicious ones. In future work we plan to use this kind of representation to automatically classify manipulations, and disambiguate legitimate cases (e.g., antivirus) from malicious ones.

IV. Limitations

We briefly discuss limitations of current REMeDy design.

First, REMeDy finds manipulations under the assumption that popular resolvers (e.g., the ISP resolvers) are legitimate. As such, if the most popular resolvers also manipulate responses, REMeDy will consider those legitimate responses and mark all remaining resolvers as anomalous. We argue that this is not a severe issue as REMeDy is designed to support the network administrators, likely aware of such situations.

REMeDy may also neglect manipulations when the legitimate and the fake server lie on the same AS. We consider this scenario unlikely, but will investigate it further in future work. Finally, REMeDy does not target some anomalies such as the hijacking of NXDOMAIN responses. We plan to extend REMeDy to detect such anomalies too.

V. Conclusions

We presented REMeDy, a system to detect manipulated DNS responses. REMeDy is a completely automatic and parameter-free system that exploits passive traffic measurements. It checks the consistency of DNS responses across multiple resolvers, and exploits anomaly detection techniques to pinpoint manipulations.

We run REMeDy on large traces collected from operational networks. It has identified 37 manipulating resolvers. We noticed a wide spread of AdWare resolvers that inject ads on victims' browsers. REMeDy evidenced also benign manipulations, used for parental control or for preventing users from contacting malicious services.

[4]More information at https://www.welivesecurity.com/2016/06/02/crouching-tiger-hidden-dns/

References

[1] M. Antonakakis, D. Dagon, X. Luo, R. Perdisci, W. Lee, and J. Bellmor. A Centralized Monitoring Infrastructure for Improving DNS Security. In *Proc. of the 13th International Conference on Recent Advances in Intrusion Detection*, pages 18–37, 2010.

[2] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster. Building a Dynamic Reputation System for DNS. In *Proc. of the 19th USENIX Security Symposium*, 2010.

[3] D. Dagon, N. Provos, C. P. Lee, and W. Lee. Corrupted DNS Resolution Paths: The Rise of a Malicious Resolution Authority. In *Proc. of the Network and Distributed System Security Symposium*, 2008.

[4] M. Kührer, T. Hupperich, J. Bushart, C. Rossow, and T. Holz. Going Wild: Large-Scale Classification of Open DNS Resolvers. In *Proc. of the 15th ACM Internet Measurement Conference*, pages 355–368, 2015.

[5] J. Laurikkala, M. Juhola, E. Kentala, N. Lavrac, S. Miksch, and B. Kavsek. Informal identification of outliers in medical data. In *Fifth International Workshop on Intelligent Data Analysis in Medicine and Pharmacology*, volume 1, pages 20–24, 2000.

[6] W. Lian, E. Rescorla, H. Shacham, and S. Savage. Measuring the practical impact of dnssec deployment. In *Proc. of the 22Nd USENIX Conference on Security*, SEC'13, pages 573–588, Berkeley, CA, USA, 2013. USENIX Association.

[7] X. Ma, J. Zhang, J. Tao, J. Li, J. Tian, and X. Guan. DNSRadar: Outsourcing Malicious Domain Detection Based on Distributed Cache-Footprints. *IEEE Transactions on Information Forensics and Security*, 9(11):1906–1921, 2014.

[8] C. Orsini, A. King, D. Giordano, V. Giotsas, and A. Dainotti. BGPStream: A Software Framework for Live and Historical BGP Data Analysis. In *Proc. of the 16th ACM Internet Measurement Conference*, pages 429–444, 2016.

[9] P. Pearce, B. Jones, F. Li, R. Ensafi, N. Feamster, N. Weaver, and V. Paxson. Global Measurement of DNS Manipulation. In *Proc. of the 26th USENIX Security Symposium*, 2017.

[10] W. Scott, T. E. Anderson, T. Kohno, and A. Krishnamurthy. Satellite: Joint Analysis of CDNs and Network-Level Interference. In *Proc. of the USENIX Annual Technical Conference*, pages 195–208, 2016.

[11] M. Trevisan, I. Drago, M. Mellia, and M. M. Munafò. Towards Web Service Classification using Addresses and DNS. In *Proc. of the 7th International Workshop on Traffic Analysis and Characterization*, pages 38–43, 2016.

[12] N. Weaver, C. Kreibich, B. Nechaev, and V. Paxson. Implications of Netalyzr's DNS Measurements. In *Proc. of the 1st Workshop on Securing and Trusting Internet Names*, 2011.

[13] C. Zhang, C. Huang, K. W. Ross, D. A. Maltz, and J. Li. Inflight Modifications of Content: Who Are the Culprits? In *Proc. of the 4th LEET*, 2011.