Going Wild: Large-Scale Classification of Open DNS Resolvers

Marc Kührer
Ruhr-University Bochum
marc.kuehrer@rub.de

Thomas Hupperich
Ruhr-University Bochum
thomas.hupperich@rub.de

Jonas Bushart
Saarland University
s9jobush@stud.uni-saarland.de

Christian Rossow
Saarland University
crossow@mmci.uni-saarland.de

Thorsten Holz
Ruhr-University Bochum
thorsten.holz@rub.de

ABSTRACT

Since several years, millions of recursive DNS resolvers are—deliberately or not—open to the public. This, however, is counter-intuitive, since the operation of such openly accessible DNS resolvers is necessary in rare cases only. Furthermore, open resolvers enable both amplification DDoS and cache snooping attacks, and can be abused by attackers in multiple other ways. We thus find open recursive DNS resolvers to remain one critical phenomenon on the Internet.

In this paper, we illuminate this phenomenon by analyzing it from two different angles. On the one hand, we study the landscape of DNS resolvers based on empirical data we collected for over a year. We analyze the changes over time and classify the resolvers according to device type and software version. On the other hand, we take the viewpoint of a client and measure the response authenticity of these resolvers. Besides legitimate redirections (e.g., to captive portals or router login pages), we find millions of resolvers to deliberately manipulate DNS resolutions (i.e., return bogus IP address information). To understand this threat in more detail, we systematically analyze non-legitimate DNS responses and reveal open DNS resolvers that manipulate DNS resolutions to censor communication channels, inject advertisements, serve malicious files, perform phishing, or redirect to other kinds of suspicious or malicious activities.

Categories and Subject Descriptors

C.2.3 [**Network Operations**]: Network monitoring

Keywords

Domain Name System; DNS Resolution Paths; Content Delivery Network; Agglomerative Hierarchical Clustering

IMC'15, October 28–30, 2015, Tokyo, Japan.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3848-6/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2815675.2815683 .

1. INTRODUCTION

The *Domain Name System* (DNS) is a hierarchical, distributed naming system and one of the core elements of the Internet. Its most common task is to provide a mapping between domain names and the associated IP addresses. As DNS is an open system, anyone is allowed to operate a publicly accessible resolver. Prior empirical studies identified millions of such resolvers that are available on the Internet [17, 27]. This, however, is counter-intuitive since the operation of publicly accessible DNS resolvers is only mandatory in rare cases only (e.g., for public DNS services such as Google DNS or OpenDNS). Furthermore, DNS resolvers are an attractive target, e.g., for cache snooping attacks [14] or as participants of amplification DDoS attacks [17, 24, 29, 37].

In 2008, an interesting attack was revealed by Dagon et al. [10], who were one of the first that documented the creation of malicious DNS resolution paths by attackers. The authors found attackers to force victims to use rogue DNS servers for all resolutions, thus being able to redirect arbitrary network connections by providing incorrect DNS replies. To understand such attacks in more detail, they performed an empirical analysis of this phenomenon: they analyzed 600,000 open DNS resolvers on the Internet and performed DNS lookup requests of 84 distinct domain names to identify unexpected, suspicious, and malicious responses.

Seven years later, this threat of open DNS resolvers continues to pose a serious problem, especially since the footprint left by attackers is small (i.e., only the clients' DNS resolution is changed). Furthermore, it is unclear how the threat landscape evolved in the last seven years. Weaver et al. provided a glimpse into this problem based on DNS traffic aggregated from Netalyzr sessions [40, 41]. Netalyzr is a Java-based applet running on end hosts of volunteers and analyzes multiple properties of the clients' Internet connections such as the reliability of the utilized DNS resolvers. In their measurement study, the authors mainly focused on DNS error monetization, a technique to redirect clients to specific advertisement websites upon requests for non-existent (`NX`) domain names. Furthermore, they also observed sessions in which DNS resolvers redirected end hosts to systems of malicious character. Yet, the number of analyzed DNS resolvers was rather low, thus it remains unclear whether the observed results generalize to all DNS servers world-wide. Several other studies analyzed DNS resolvers [1, 4, 25, 33, 34], yet these studies were mainly conducted on a small subset of all resolvers in the IPv4 address space. As such, open DNS resolvers and the potential of malicious DNS resolution paths remain an underexplored phenomenon on the Internet.

In this paper, we illuminate this phenomenon and present the results of a long-term, large-scale empirical study of DNS resolvers on the Internet. In a first step, we monitor the landscape of devices that listen to DNS requests and reply with valid DNS responses. Based on the data collected via weekly Internet-wide scanning activities for more than one year, we provide a comprehensive overview of the resolvers' landscape. During our 13-month-long study, the number of DNS resolvers dropped from 26.8 to 17.8 million servers. To get a better understanding of these hosts, we analyze their properties in terms of geographical and network-based distribution, operated DNS server software, and underlying hardware. We find 76.4% of the resolvers in the Top 25 networks (that operate most servers) to be associated with various broadband telecommunication providers world-wide.

In the second phase of our study, we take the viewpoint of a client system to analyze the *integrity* of the DNS resolutions provided by open recursive DNS resolvers. That is, we aim to understand whether the resolvers actually return legitimate answers or perform bogus resolutions to provide clients with false IP address information. To this extent, we manually selected a set of 155 domains from 13 different website categories (i.e., banking pages, ad providers, adult sites, etc.) that an attacker potentially wants to tamper with. We then perform DNS lookup requests for all domain names on each identified open DNS resolver in the IPv4 address space. By prefiltering "correct" pairs of resolvers and the IP addresses they returned, we sort out all previously seen legitimate answers to significantly reduce the data set. The remaining answers are potentially incorrect, hence we aim to understand them in more detail. We thus request HTTP content for each non-legitimate DNS response and cluster the responses by leveraging agglomerative hierarchical clustering. In the final step, a manual labeling of these clusters enables us to understand the nature of bogus resolutions. Besides legitimate redirects (e.g., captive portals), we find many cases of censorship, ad redirections, and suspicious activities. More specifically, more than 3 million resolvers redirect requests of specific domain names to a small set of IP addresses that host landing pages for censorship of 34 countries. Our in-depth analysis of the HTTP content also enables us to reveal other kinds of abuses. We find 281 resolvers to redirect ad traffic of two large ad providers, while 228 DNS resolvers redirect client systems to malicious content. Further 10,179 resolvers returned a set of IP addresses that act as HTTP proxies for all requested domains. As such, clients might risk to disclose sensible login information when relying on these open recursive DNS resolvers.

To summarize, our contributions are as follows:

- We perform a long-term, large-scale empirical monitoring of DNS resolvers on the Internet, study the changes over time, and classify the resolvers based on characteristics such as device type and software version.
- We systematically evaluate the authenticity of DNS responses when querying open recursive DNS resolvers. That is, we study whether DNS resolvers return incorrect responses upon requesting a set of 155 domain names from 13 different categories, whereas one main focus of our analyses is DNS-based censorship.
- We study in detail the content returned by manipulated DNS responses from open DNS resolvers and uncover different cases of fraudulent manipulation, respectively, phishing of web content and email traffic.

Domains and Datasets. The complete list of scanned domains can be found at `http://syssec.rub.de/research/dns`. Upon request, we further provide access to all datasets that we addressed throughout our analyses in this paper.

2. DNS RESOLVERS IN THE WILD

We begin with a general overview of DNS resolvers in the IPv4 address space. That is, we provide information regarding the magnitude of systems responding to DNS requests and aim to shed light onto the deployed software versions and hardware devices of these hosts. We then discuss our findings in terms of IP address churn and analyze the utilization of the identified DNS resolvers in more detail.

2.1 Terminology

Before going into our analyses, we briefly introduce the DNS-specific terminology that we use throughout this paper. We distinguish between *recursive resolvers* (or "resolvers") and *authoritative name servers* (or "AuthNS"). When requesting the resolution of a domain, a resolver follows the domain name hierarchy and iteratively contacts the AuthNSes of the domain to look up a DNS request. In contrast, an AuthNS is responsible for answering lookup requests for its *zone*, i.e., a subset of the DNS hierarchy (e.g., a domain). An AuthNS does not need to process lookup requests for domains other than in its zone, hence we strictly separate AuthNSes from resolvers (although we may find hosts that serve both roles). For resolvers, we further distinguish between *open* resolvers, which accept and process DNS queries from any external location, and *closed* DNS resolvers, which allow queries from a trusted subset of IP addresses only.

2.2 Resolver Magnitude

First, we monitor the landscape of devices that listen to DNS requests and reply with valid DNS responses. That is, we perform weekly Internet-wide scans in IPv4 for more than one year and enumerate the responsive DNS resolvers.

Scanning Setup. To perform Internet-wide scans in the IPv4 address space, we implemented an efficient scanning software. We adopted multiple scanning practices suggested by Durumeric et al. [12] to abide reasonable scanning behavior. For example, our scanner applies a linear feedback shift register (LFSR) of order $2^{32} - 1$ to distribute the sequence of target IP addresses. As such, scanned networks receive a limited number of DNS requests within a short time frame.

In an IPv4 scan, we send a single DNS `A` lookup request for a specific domain to each IP address in IPv4, excluding well-known private and unallocated network ranges. Each request is specifically crafted and includes a random domain prefix (to avoid caching) and the hex-formatted IP address of the target host in the form `prefix.hex-ip.domain.edu`. This allows us to obtain the IP address of the target host to which the request was sent by inspecting the DNS response.

To offer networks to opt-out from our scanning activities, we defined a reverse DNS (rDNS) record for the scanning system and set up a web server with project details. In total, we added 208 network ranges and 50 individual IP addresses to our blacklist upon request of a network administrator,

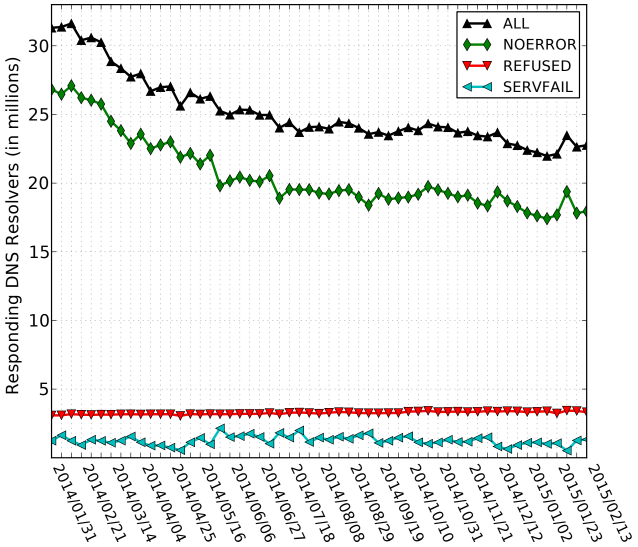


Figure 1: DNS resolvers identified in our weekly scans

resulting in 20,834,166 blacklisted IP addresses. To allow comparisons between the individual weekly scans, we ignore blacklisted IP addresses in all of our scanning results.

Results. We initiated the first DNS scan on Jan 31, 2014 and performed weekly scans for more than one year. Figure 1 outlines the total number of unique IP addresses responding to our requests and illustrates the number of systems that replied with the most common DNS status codes (`NOERROR`, `REFUSED`, and `SERVFAIL`). Note that `NOERROR` includes all hosts that sent a DNS response with this status flag set—regardless of the actual content of the DNS answer section. That is, we consider resolvers that returned legitimate IP addresses for the requested domain as well as hosts that specified empty answer sections or bogus `A` records.

At the beginning of our scanning activities, we find more than 26.8 million systems to respond with `NOERROR`. The number of resolvers drops to 17.8 million IP addresses over time, a phenomenon we address later on. Resolvers specifying the `REFUSED` error flag remain stable during our monitoring, whereas the number of systems returning `SERVFAIL` fluctuates between 633,393 on Dec 19, 2014 and 2,141,539 on May 30, 2014. For the remaining error codes (e.g., `NXDOMAIN`), we find a negligible number of DNS servers only.

When comparing the source IP addresses in the UDP packets with the target addresses that were encoded in the requested domain names, we observe 630,000 to 750,000 resolvers per week to respond to DNS requests that were sent to different target hosts. Prior work has shown that these resolvers are either multi-homed systems or DNS proxies that forward DNS requests to different recursive resolvers [17].

Scan Verification. We further estimate the effect of networks blocking our periodic DNS requests. We thus performed an Internet-wide DNS scan from a secondary host in a different /8 network and compared the number of responsive systems. We find 692,283 DNS resolvers to respond to our verification scan but not to the corresponding weekly scan. 482,158 of these hosts (69.6%) returned `SERVFAIL`, while 64,821 systems (9.4%) refused our DNS request. Accordingly, we missed 145,304 resolvers replying with `NOERROR` in our weekly scan, which is $<$ 1% of all identified resolvers.

Table 1: Resolver fluctuation per country

	Resolvers (in #)		Fluctuation	
Country	Jan 31, 2014	Feb 06, 2015	(in #)	(in %)
US	2,958,640	2,537,269	-421,371	-14.2
CN	2,418,949	2,104,663	-314,286	-13.0
TR	1,439,736	976,226	-463,509	-32.2
VN	1,393,618	1,039,075	-354,543	-25.4
MX	1,372,934	1,175,343	-197,591	-14.4
IN	1,269,714	1,431,522	+161,808	+12.7
TH	1,214,042	564,482	-649,560	-53.5
IT	1,172,001	722,756	-449,245	-38.3
CO	1,062,080	677,572	-384,508	-36.2
TW	1,061,218	453,016	-608,202	-57.3

Table 2: Resolver fluctuation per *Regional Internet Registry*

	Resolvers (in #)		Fluctuation	
RIR	Jan 31, 2014	Feb 06, 2015	(in #)	(in %)
RIPE	11,193,636	7,475,795	-3,717,841	-33.2
APNIC	10,431,352	7,878,208	-2,553,144	-24.5
LACNIC	5,136,320	3,335,895	-1,800,425	-35.1
ARIN	3,143,388	2,761,875	-381,513	-12.1
AFRINIC	1,305,747	1,193,178	-112,569	-8.6

We also compared our numbers of active resolvers to those provided by the *Open Resolver Project* [27] and find the numbers for each scan to match within a 2% error margin.

2.3 Geographical Distribution

After observing the resolvers' landscape, we shed light onto the geographical and network-based distribution.

GeoIP-based Statistics. We first determine the geographical distribution of the resolvers using the MaxMind GeoIP database [26]. That is, we enumerate the number of DNS resolvers per country at the beginning and the end of our scans and determine the fluctuation of resolvers. Table 1 outlines the Top 10 countries hosting 49.1% of all DNS resolvers on Jan 31, 2014 and their fluctuation within one year. During our measurement study, the number of resolvers decreased for most countries. The highest decrease is found for Argentina with -75.0%, while the numbers also dropped significantly for Great Britain (-63.6%) as well as Taiwan (-57.3%). We further identify six countries in which all DNS resolvers (up to 63 hosts per country) vanished.

While the number of resolvers declined for many countries, we also observe opposite behavior. Besides India with an increase of +12.7%, also Malaysia and Lebanon had significantly more resolvers (+59.7% and +76.7%, respectively).

Table 2 outlines the resolver distribution per *Regional Internet Registry*. The decrease in resolvers is not seen for a single registry only, yet can be observed best for RIPE, APNIC, and LACNIC, which (roughly) represent the regions Latin America and Caribbean, Europe, and Asia-Pacific.

AS-based Statistics. We now analyze the Autonomous Systems (AS) distribution of the DNS resolvers to get a more detailed understanding what might have caused the drop in resolvers during our scanning activities. We find the highest decrease of 720,843 resolvers (-97.8%) in the network of an Argentinean telecommunication provider, primarily causing the high decrease of DNS resolvers in Argentina. While this network operated 737,424 resolvers in Jan 2014, we find less than 17,000 systems in 2015. We observe similar results for a South Korean Internet service provider (ISP): 434,567 resolvers replied to our initial scan in Jan 2014, whereas we find only 22 DNS resolvers in the last weekly IPv4 scan.

In total, 28 networks operated more than 1,000 DNS resolvers in Jan 2014 (76,973 hosts in total, i.e., 0.2% of all identified resolvers in Jan 2014), however, not a single active resolver at the end of our measurement study. There are three possible explanations: (i) our requests were blocked at the network level during the measurement period, (ii) DNS egress or ingress filtering was added to the network, or (iii) all actively operated DNS resolvers have been shut down.

Explanation (i) holds for 21 networks, as we still observe active DNS resolvers in our verification scan. To distinguish between the other two explanations, we monitor the fluctuation of resolvers for each weekly scan. If a network operates ≥ 100 resolvers and we cannot find a single active resolver in the subsequent week, we assume this network to apply DNS filtering. For networks that have less than 100 DNS resolvers, we assume explanation (iii) to hold. For the remaining seven networks, five networks perform DNS filtering, while two seem to have shut down all servers. Note that we chose these thresholds to allow comparisons with the decrease in NTP amplifiers addressed by Kührer et al. [17].

For the Top 25 networks that include most of the DNS resolvers in Feb 2015, at least 20 networks offer end user services such as telecommunication and Internet via broadband. While this does not necessarily imply that all DNS resolvers in these networks are running on consumer devices such as modems and routers, it is a first indication that at least a set of DNS servers might be running on devices of broadband customers—potentially unsupervised or operating unknowingly by the end users. We thus assume that a certain number of vanishing DNS resolvers is caused either (i) by ISPs when introducing egress or ingress filtering of DNS traffic for their customer IP spaces or (ii) by software updates (such as on home routing equipment via TR-069 [8]) that—among other bug fixes—also restricted the accessibility of the DNS stub resolvers from outside the local network.

2.4 Resolver Classification

Next, we classify the resolvers according to the operated server software and the underlying hardware specifications.

Fingerprinting DNS Server Software. On Dec 17, 2014, we initiated an IPv4 scan using CHAOS [13] `version.bind` and `version.server` requests to identify the DNS server software and obtained responses from 19,925,818 open resolvers, of which 42.7% replied with error codes (`REFUSED` or `SERVFAIL`) for both version requests. Further 4.6% replied with `NOERROR`, however, did not specify any version in both responses. Additional 18.8% of the responding systems returned arbitrary version strings that were specified by administrators to hide software information. As such, two thirds of the DNS resolvers did not leak software details.

For the remaining 33.9% of open resolvers (i.e., 6,753,748 systems), we could obtain software and version information. Table 3 illustrates the Top 10 used server software among the responses that include version details. The majority of these resolvers (60.2%) operate *BIND* (i.e., at least 20.4% of all identified DNS resolvers). Alarmingly, not all operate the newest *BIND* versions, though. For example, about 20% of the resolvers run *BIND* 9.8.2, which is known to be vulnerable to memory-exhaustion attacks. Furthermore, two versions of *BIND* (23.7%) are prone to IP range bypassing, enabling remote attackers to circumvent IP address restrictions and perform requests at closed DNS resolvers. In addition, all Top 10 software versions are susceptible to DoS attacks that can crash the operated DNS server software.

Table 3: Results for the CHAOS version requests based on the 6,753,748 DNS resolvers returning version information

Software	Resolvers	Released	Deprecated	CVE
BIND 9.8.2	19.8 %	Apr 2012	May 2012	IP Bypass, DoS Mem. Corr./Leak.
BIND 9.3.6	8.9 %	Nov 2008	Jan 2009	DoS
BIND 9.7.3	5.7 %	Feb 2012	Nov 2012	Mem. Overfl., DoS
BIND 9.9.5	5.2 %	Feb 2014	Sep 2014	DoS
Unbound 1.4.22	4.8 %	Mar 2014	Nov 2014	Mem. Overfl., DoS
Dnsmasq 2.40	4.6 %	Aug 2007	Feb 2008	RCE, DoS
BIND 9.8.4	3.9 %	Oct 2012	May 2013	IP Bypass, DoS Mem. Overfl.
PowerDNS 3.5.3	3.2 %	Sep 2013	Jun 2014	DoS
Dnsmasq 2.52	2.9 %	Jan 2010	Jun 2010	DoS
MS DNS 6.1.7601	2.5 %	Jun 2011	Aug 2011	DoS

Fingerprinting Hardware Devices. Our next goal was to obtain more detailed system information about the individual resolvers such as the OS and the hardware specifications of the underlying device. The DNS protocol itself does not provide any of this information. As such, we rely on system information provided by other services that might be running on a resolver. That is, we initiate FTP, HTTP, HTTPS, SSH, and Telnet connections and analyze potentially returned banner information and text fragments to fingerprint the device [17]. Upon our connection attempts, we obtained payload data for at least one TCP protocol for 26.3% of the responsive DNS resolvers (i.e., 5,459,524 systems). The remaining resolvers (15,307,037 servers in total) did not offer any public TCP services for the scanned protocols that we could leverage for the device fingerprinting.

To generate fine-granular fingerprints, we manually compiled more than 2,245 regular expressions by closely analyzing the aggregated responses for tokens such as device identifiers. We then aimed to find more specific information about each device type and leveraged manuals and online specifications to attribute details for fingerprinting. The token "dm500plus login", for example, is associated with a DVR running a Linux-based OS on a PowerPC architecture.

Table 4 illustrates the device fingerprinting results. We group routers, modems, and gateways in one category, as these devices often provide overlapping functionality. Category *Embedded* includes devices which we find to run embedded OSes or applications (e.g., the web servers *GoAhead-Webs* and *RomPager*) but lack information to fingerprint the hardware more precisely. This category further includes devices such as *Serial to LAN* converters or micro controller boards such as *Arduino* and *Raspberry Pi*. We find 34.1% of the DNS resolvers responding to our TCP requests to be business and consumer routing devices. In particular, we find three major manufacturers of consumer broadband devices to be prevalent. For example, *ZyNOS*, an OS deployed on devices of manufacturer *ZyXEL*, runs on 16.6% of the DNS resolvers. Besides the large group of routing equipment, we find smaller clusters such as IP-based cameras and DVRs as well as 10,962 NAS devices and 5,061 DSLAMs, operated by ISPs to provide DSL support to their customers.

2.5 IP Address Churn of Resolvers

Seeing a vast number of DNS resolvers to run on routing equipment, we next evaluate the churn of IP addresses of these systems. That is, we estimate the interval between

Table 4: Device fingerprinting results of the 5,459,524 DNS resolvers responding to our TCP-based requests

	Hardware (in %)							Operating System (in %)								
	Router	Embedded	Firewall	Camera	DVR	Others	Unknown	Linux	Unix	ZyNOS	Windows	SmartWare	RouterOS	CentOS	Others	Unknown
DNS	34.1	30.6	1.9	1.8	1.2	1.1	29.3	23.2	21.3	16.6	5.0	3.6	2.6	1.7	2.1	23.9

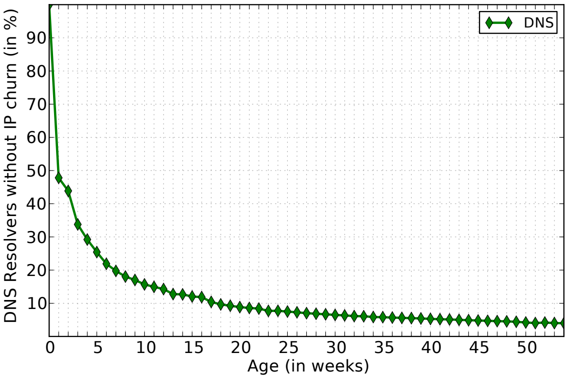


Figure 2: IP address churn of DNS resolvers for 55 weeks

observing a DNS resolver in our Internet-wide scans and the resolver changing its IP address. To this end, we enumerate the DNS resolvers that we identified in our first scan on Jan 31, 2014 based on the observed IP addresses. In the subsequent weeks, we probe these 26,820,486 servers to check if they continuously provide DNS resolutions over time.

Figure 2 outlines the ratio of DNS resolvers that remain stable in terms of their IP addresses during our scanning activities. The initial IP address churn is quite high—52.2% of the DNS servers disappear within a single week. After one year, 1,073,211 systems (i.e., 4.0% of the initially discovered DNS resolvers) still offer DNS resolutions. A fifth (19.7%) of these systems is operated in networks of a Colombian (9.5%), respectively, Turkish (5.3%) telecommunication provider and a US hosting company (4.9%), while the remaining resolvers are widely distributed in terms of ASes.

Observing the high IP address churn within the first week, we measure when exactly the resolvers changed their IP addresses. In fact, more than 40% disappear within the first day—an indicator for DNS resolvers running on devices with low IP lease times such as consumer routing devices. To verify our assumption, we aggregate rDNS records for all resolvers that disappeared after one day and match the records against tokens indicating dynamic IP address assignment (e.g., `broadband`, `dialup`, and `dynamic`). Indeed, at least 67.4% of the 1,989,502 IP addresses we find to provide rDNS records are assigned to dynamic broadband Internet links.

2.6 Resolver Utilization

Faced with the high number of DNS resolvers we wondered if these resolvers are in use at all, hence providing DNS services to actual clients. We therefore analyzed the resolvers' caches using DNS cache snooping [14]. That is, we requested name server (`NS`) records for 15 Top Level Domains (TLDs) (i.e., *br*, *cn*, *co.uk*, *com*, *de*, *fr*, *in*, *info*, *it*, *jp*, *net*, *nl*, *org*, *pl*, and *ru*) every 60 minutes for 36 hours to monitor the associated Time To Live (TTL) values. We then checked if these TLDs were re-added to the resolvers' caches after their expiration. This indicates that a client performed DNS requests for domains associated with these TLDs, causing a resolver to request information from the corresponding AuthNSes. We do not expect to aggregate results from each DNS resolver for the whole time due to IP address churn. Yet, caching values for a few hours might already reveal the desired details about a resolver's usage.

On Nov 30, 2014, we initiated an Internet-wide scan to identify the active resolvers and successively performed the monitoring of the resolvers' caches. Of the identified resolvers, 83.2% (i.e., 13,214,020 systems) respond to at least one of our DNS cache snooping requests, while we suspect the remaining hosts to become unreachable due to IP churn in the mean-time. We find 7.3% of all resolvers to reply with empty DNS responses instead of `NS` records. Additional 3.3% of the hosts send a single response for each TLD before stopping to reply—presumably due to IP churn. For 4.0% of the DNS resolvers we obtain either a static TTL value for each `NS` request or the TTL was set to value 0.

For the remaining resolvers, we flag a resolver as *in use* when we indeed observe TLDs getting re-added to the cache after their expiration. To verify that our results are not biased due to Internet-wide scans performed by other organizations or research projects, we monitored incoming DNS requests to our network and identified a small set of hosts that performed DNS scanning activities. That is, we observe a single daily DNS request initiated by the *Shadowserver Foundation* and single DNS requests from *Team Cymru* and the *Open Resolver Project* within our monitoring period of 10 days. These scans performed `A` lookup requests for three domains assigned to two distinct TLDs. To ensure that at least one TLD cache refresh was initiated by a real client, we require at least three TLDs to be re-added to a resolver's cache during our monitoring to flag a resolver as *in use*. In total, we find 61.6% of all resolvers (i.e., 9,788,740 hosts) to be actively used, i.e., we find at least three TLDs that are refreshed during our monitoring. 6,142,328 systems (i.e., 38.7% of identified DNS resolvers) seem to be used frequently by clients, as we find at least one TLD to be re-added to the resolvers' cache within 5 seconds after expiration. For 4.0% of all DNS resolvers, we monitor a decreasing TTL value for each `NS` request, yet did not obtain sufficient responses to observe the caching entries to expire. Another 19.6% kept resetting the TTL values way ahead before expiration. We assume that many of these resolvers either (i) reset TTL values proactively before expiration or (ii) are associated with groups, connected via load balancers. Depending on which DNS resolver handles the hourly request, we obtain caching values from different DNS server caches.

As a follow-up of our work, one can use a more fine-grained DNS cache snooping technique to evaluate the time gap be-

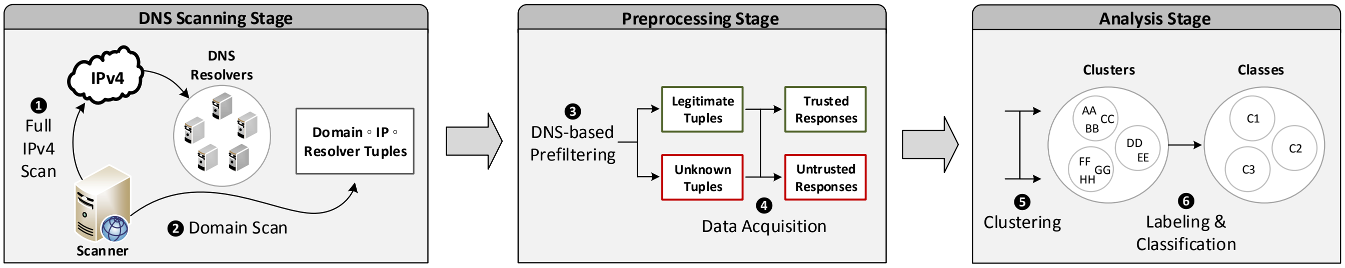


Figure 3: Processing chain to identify manipulated DNS resolutions

tween recaching entries, aiming to approximate the *popularity* of open resolvers, as suggested by Rajab et al. [28].

3. MANIPULATED DNS RESOLUTIONS

So far, we found millions of DNS resolvers that pose a potential security threat. Not only that many of these systems are potentially susceptible to exploitation due to outdated DNS server software, but DNS servers are also known to significantly contribute to large-scale amplification DDoS attacks [17, 24, 29, 37], such as the 300 Gbps attack against Spamhaus in 2013 [23]. However, the vast number of open resolvers also allows us to dig closer into the behavior of DNS resolutions world-wide. In particular, we are interested to understand the *integrity* of the DNS resolutions. We thus ask the following research question: do the open resolvers actually operate *correctly*, i.e., do they strictly follow the clearly-defined hierarchy in the Domain Name System?

We use the processing chain illustrated in Figure 3 to answer this question. First, we identify open DNS resolvers in an Internet-wide scan (step 1) and query them for a manually-chosen list of domain names of various categories (step 2). Afterwards, we perform a prefiltering process to sort out the vast majority of "correct" DNS responses, i.e., we filter ($domain \circ ip \circ resolver$) tuples for all DNS resolver and domain combinations for which we find the resolver to return legitimate IP addresses (step 3). Accordingly, all non-filtered (i.e., unexpected) responses are considered as suspicious and grouped as "unknown". For further analyses and classification—particularly of the unexpected DNS responses—we request HTTP(S) content (and for particular domain names IMAP, POP3, and SMTP banner information) for all legitimate and unknown ($domain \circ ip \circ resolver$) tuples in step 4. In step 5, we cluster the unexpected responses, while we manually label the resulting groups in step 6 to classify virtually all received responses. The following subsections describe these processing steps in more detail.

Note that the DNS protocol offers various features and record types, yet in this section we focus on name resolutions in the IPv4 address space (i.e., `A` records that return IP address information upon requesting a domain name).

3.1 Threat Model

Our threat model affects clients that use and blindly trust DNS resolvers, which may or may not return "correct" recursive DNS resolutions. With "correct" we refer to DNS lookups that strictly follow the hierarchy, i.e., starting at the root (`.`), resolving the TLD (e.g., `.com.`), and then iteratively querying the AuthNSes of a domain name to resolve the fully-qualified domain (e.g., `example.com.`). In principle, we suspect the majority of actively operated recursive DNS resolvers to follow this general resolution procedure, yet prior work [10, 40, 41] already revealed that adversaries tend to deploy bogus resolvers to return non-legitimate responses. This is a severe attack, as bogus resolvers have almost complete control over the network traffic of their clients, thus can redirect the end hosts to arbitrary IP addresses for the requested domains. In our threat model, we thus focus on resolvers that do *not* follow the general DNS hierarchy but respond with forged DNS resolutions. So far, it is unclear to which extent these fraudulent DNS resolvers are actively operated in the entire IPv4 address space.

Note that the threat of incorrect DNS resolutions is not specific to *open* DNS resolvers only and may similarly affect correctly-protected resolvers (i.e., DNS resolvers that only respond to lookup requests of particular IP addresses). In this paper, however, we focus on open resolvers, as they are reachable to the public, and thus allow us to perform large-scale measurements of the DNS resolutions world-wide.

Furthermore, any client system that utilizes open DNS resolvers is potentially threatened by bogus DNS resolutions. This does not only affect the presumably few people who deliberately choose these servers as their personal DNS resolvers. In fact, as we have seen, many resolvers operate on networking equipment, indicating that these systems—by default—have end users who use the provided DNS services.

In our threat model, we do not make any assumption on specific threats that clients may face when relying on incorrect DNS responses. Instead, we systematically analyze the responses and categorize the monitored threats. In general, we can only speculate on the reasons why DNS resolvers turn wild and return unexpected responses. For example, DNS servers might simply be incorrectly configured or return IP addresses of captive portals to redirect client systems to the login page of a (e.g., wireless) network. Furthermore, resolvers can redirect users to search websites for misspelled or non-existing domain names. However, there are also more severe reasons for abnormal DNS responses, e.g., censorship to restrict the access to certain web pages (such as adult content or social networks). Likewise, attackers might aim to sniff on the clients' Internet traffic by transparently proxying the communication to the original websites. Similarly, we imagine attackers that slightly manipulate the web content, e.g., in order to inject advertisements or malicious code.

3.2 Datasets

For our analysis, we determine the response behavior for various types of domain names and check whether we find incorrect DNS responses. Assuming a malicious DNS resolver, it is unclear for which particular domain names it manipulates the answers. Clearly, in order to scale, we cannot exhaustively query thousands of domains at millions of resolvers. We therefore choose a reasonable domain set by selecting a few domain names from different categories. Based on our threat model, we select domain names for various categories that could be relevant for DNS response forgery. For example, we selected banking domains that were targeted by banking trojans via web injects in our dynamic malware analysis platform SANDNET [30]. Furthermore, we include DNS protection services such as operated by antivirus companies, which alter DNS responses for domains associated with prevalent malware families, while the chosen dating and gambling domains are censored in various countries. In total, our domain set consists of 155 domain names, grouped in the following 13 website categories:

- *Ads*: 9 domains associated with ad providers.
- *Adult*: 4 domains taken from the *Alexa Traffic Ranking* [2] that provide adult content.
- *Alexa*: Top 20 ranked domains taken from Alexa.
- *Antivirus*: 15 domains of web pages and update servers of AV and malware protection companies.
- *Banking*: 20 domain names of banking and credit card websites. This includes popular online payment websites such as Alipay, Ebay, and PayPal.
- *Dating*: 3 domains of dating sites, e.g., `match.com`.
- *Filesharing*: 5 domain names of file sharing websites, e.g., `kickass.to` and `thepiratebay.se`.
- *Gambling*: 4 domains associated with online betting and gambling such as `bet-at-home.com`.
- *Malware*: 13 domains associated with malicious activities that are knowingly listed by many common malware blacklists [18], e.g., `irc.zief.pl`.
- *MX*: 13 hostnames of IMAP, POP3, and SMTP servers of 6 organizations offering mail services (Aim, Gmail, Mail.me, Outlook, Yahoo, Yandex).
- *NX*: 8 domain names that are non-existent, 5 NX subdomains of popular domains such as `rswkllf.twitter.com`, and 8 NX domains that include spelling mistakes or missing/additional letters or permutations such as `amason`, `ghoogle`, and `wikipeida`.
- *Tracking*: 5 domains of user tracking libraries such as operated by BlueCava and ThreatMetrix.
- *Miscellaneous*: 6 domains of update servers (e.g., Adobe and Windows), 3 domains of intelligence agencies (NSA, GCHQ, and Mossad), 3 domains of OAuth services (Amazon, Google, and Twitter), and 10 individual domains such as `rotten.com` and `wikileaks.org`.

Finally, we resolve a domain name, for which we operate the AuthNSes. This domain serves as ground truth (GT) to identify DNS servers that return legitimate IP addresses for the GT domain but bogus or empty DNS responses for the remaining domains in the above-mentioned domain sets.

3.3 Domain Scanning Setup

For each of the domain categories, we perform an individual scanning process to obtain DNS responses from all open resolvers. The scanning setup resembles the setup that we discussed in Section 2. Yet, instead of probing all IPv4 addresses, we send DNS `A` lookup requests for all domains in a particular domain set (plus the GT domain) to only those open resolvers that we identified as such by an Internet-wide scan right before initiating the domain scan. We again employ an LFSR and highly reduce the scanning speed in order to distribute and limit the load both on the individual DNS resolvers and on the AuthNSes of the scanned domains.

We again encode the IP address of each target host (i.e., resolver) in the DNS request to differentiate between DNS responses that specify the same source IP address in the UDP packet, although the actual request was sent to a different target host. However, we cannot specify the target IP directly in the queried domain name as conducted in Section 2, as the set of domains is fixed. Instead, we assign each previously identified open resolver a unique identifier. As we find up to 20 million open resolvers in the weekly IPv4 scans in 2015, we have to store $\lceil log_2(20,000,000) \rceil = 25$ bits of information in each request, reducing the number of required bits to encode an IP address from 2^{32} to 2^{25}. We encode 16 bits of information in the DNS transaction ID of the request, while the remaining 9 bits are encoded in the UDP source port (i.e., we send packets using 2^9 different ports). To render redundancy—some resolvers change the destination port of the DNS response for some reason—we encode the 9 bits also in the domain name using `0x20` encoding [9] (i.e., setting specific characters to upper and lower case).

3.4 DNS-based Prefiltering

When initiating DNS `A` lookup requests for 155 domains at 20 million resolvers, we are consequently flooded with billions of DNS responses. As we do not expect the majority of DNS servers to provide forged responses, we aim to filter the vast majority of legitimate answers. As such, we apply a heuristic to determine the legitimacy of each IP address in every DNS response. Note that at this stage we do not want to risk filtering bogus responses but can tolerate to not filter legitimate responses, which we can filter in a later step.

As long as domains are assigned to a fixed set of IP addresses, identifying valid DNS responses is quite easy. However, load balancing and *Content Delivery Networks* (CDNs) may lead to the effect that domains point to thousands of distinct IP addresses and answers may vary on the resolvers' geographical location. To complicate filtering further, IP addresses assigned to a domain name may span multiple ASes. As we request domain information from DNS resolvers located at various places world-wide, we thus likely obtain large sets of IP addresses for particular domain names.

To filter legitimate DNS responses and reduce the initial set of replies to a reasonable size, we apply multiple filtering methods. For *non-existing* domains, we filter responses that either return `NXDOMAIN` or return `NOERROR` without specifying any IP addresses in the answer section. DNS responses for *existing* domain names are filtered when we find each returned IP address to match one of the two following criteria:

(i) We perform a DNS `A` lookup at (trusted) recursive resolvers and find the IP in question to be located in the same ASes as the IP addresses we resolved ourself.

(ii) A rDNS record is assigned to the IP address. If the domain part of the record resembles the requested domain, we filter the IP address. However, as rDNS records can be set to arbitrary domains even without owning the domain, we further require that an `A` lookup request of the rDNS record returns the IP in question, as only the domain owner can set up the `A` record.

Using these filtering rules, we identify legitimate IP addresses for domains that are located in one or a few networks only. Yet, we miss IP addresses of CDN providers that are not located in the providers' primary networks but distributed in various networks world-wide. For example, Akamai is directly associated with at least 8 ASes, yet also distributes their content in several other ASes. In order to find IP addresses that belong to CDN providers, we request HTTPS certificate information at all previously-unfiltered IP addresses using the domains for which these IP addresses were returned. More specifically, we perform two HTTPS requests for each ($domain \circ ip$) pair we find in ($domain \circ ip \circ resolver$) tuples—the first with the TLS *Server Name Indication* (SNI) extension enabled, while we disable SNI in the second request to also obtain the default certificate delivered by a web server. We only consider an IP to be legitimate, if a valid and known certificate was returned for the requested domain name. For the largest CDN providers, we further consider an IP address to be valid, if the non-SNI certificate is legitimate and includes a particular common name.

Note that we might not identify each legitimate DNS response for every queried domain name, as some IP addresses could still be valid, even though none of our filtering rules applied. Furthermore, we do not want to risk filtering bogus DNS responses but can tolerate to not filter legitimate responses. As such, the resulting dataset of "unknown" DNS responses might also include legitimate IPs, yet these can be filtered in a later step (i.e., by analyzing the served content).

3.5 Data Acquisition

Prefiltering supports us in eliminating the vast majority of legitimate responses. For all remaining (i.e., mostly unexpected) responses, we mimic a client system that utilizes the "abnormal" resolvers to aggregate data for further analyses.

To aggregate HTTP data, we use the IPs returned by the resolvers upon resolving the domains and request HTTP information as if they belong to the original website that we tried to contact. That is, we request HTTP and HTTPS content by impersonating a client using the Mozilla Firefox web browser (version 28.0) for all ($domain \circ ip \circ resolver$) tuples we could not classify as legitimate during our prefiltering phase. If HTTP content includes redirections or frames (i.e., an `<iframe>`), we follow these redirections two times at most. If content redirects to further (sub-)domains, we resolve these new domain names at the DNS resolver that returned the initial ($domain \circ ip \circ resolver$) tuple. Note that our succeeding analyses are performed on the plain HTTP(S) payload data, i.e., we do not execute JavaScript.

For the *MX* domain set (cf. Section 3.2), we further initiate IMAP, POP3, and SMTP connections to the respective IP addresses to aggregate banner information for analysis.

We furthermore obtain a list of ($domain \circ ip \circ resolver$) tuples from our own (trusted) DNS resolvers for all domain names in the scanned domain set. Similar to the previous steps, we aggregate HTTP(S) (and IMAP, POP3, and SMTP) content for this legitimate dataset. The resulting payload data serves as ground truth to allow comparisons with potentially-malicious and benign responses.

3.6 HTTP Data Analysis

The HTTP data acquisition leaves us with huge datasets for further analysis. We aim to analyze the aggregated payload data as exhaustively as possible, however, given millions of HTTP responses, this is not feasible using a purely-manual process. We therefore use unsupervised learning to group similar responses. Later on, we manually analyze the resulting groups and augment them with descriptive labels. In the following, we describe this procedure in more detail.

Coarse-Grained Clustering: Grouping Similar HTTP Responses. In a first step, we aim to shrink the scale of the HTTP payload data that we need to manually inspect. We decided to use agglomerative hierarchical clustering, an unsupervised machine learning technique, which helps us to group similar responses into *clusters*. Hierarchical clustering is known to be helpful for inspecting the clustering results (e.g., using the dendrograms) and enables the analyst to understand the clustering steps (e.g., via the distance matrix).

In addition, hierarchical clustering allows us to define a custom distance function to measure the similarity of two individual HTTP responses. We deploy seven normalized features of equal weight as part of our distance function:

- Length difference of the HTTP response body as a first coarse-grained comparison feature.
- Jaccard distance (i.e., $\frac{A \cap B}{A \cup B}$) for multisets with the set of HTML tags in the HTTP payload.
- Edit distance between the sequence of opening HTML tags, whereas we convert each HTML tag to a 2-byte-long identifier to normalize all tags. In contrast to the previous tag multiset comparison, this feature also considers the order of the elements. It reflects the structural similarity between two web pages
- Edit distance on the `<title>` value. We focus on the title only for content comparison as it is rather constant. We do not consider any other texts, as these could highly differ for dynamic web pages.
- Edit distance of all JavaScript code. The motivation here is that many websites largely rely on JavaScript (e.g., AJAX) to control the content of the page.
- Lastly, each individual feature focuses on (i) embedded resources and (ii) outgoing links, respectively. Resources and links can be used to attack or track visitors. We compute the Jaccard distance for all embedded resources (i.e., the values of `src=""` attributes), and for outgoing links (i.e., the values of `href=""` attributes), respectively.

Clustering assists us in grouping similar representations of certain web pages, deliberately ignoring most of the (possibly dynamic) HTML content. In addition, clustering is tolerant to slight changes in the page structure. Similar instances are grouped using average linkage. This way, we can label all instances in a cluster once the cluster has been identified, largely reducing the manual inspection effort.

Fine-Grained Clustering: Finding Page Modifications. While our coarse-grained clustering step is helpful to group similar websites, it also abstracts from slight modifications in the HTML structure. This can be problematic, as we are interested in cases where a potential adversary uses a known representation of a website and modifies it. Therefore, we also use clustering to analyze the difference of websites compared to the respective legitimate representations (i.e., comparisons to the *ground truth*). The key idea is to find small, possibly malicious modifications (e.g., JavaScript injections) that were done to the original representation.

To this end, we first use the `diff` utility to find the exact differences between an unknown response and the legitimate representation(s) of a website. If we have multiple legitimate representations, we select the ground truth with the highest similarity for further processing. From the exact difference, we then extract which HTML tags were added and removed—the smaller these sets, the fewer modifications were done to the website. Again, we aim to group responses, but this time we are interested in similar *modifications*. We thus use hierarchical clustering based on the Jaccard distance for multisets for the tag differences between the unknown response and the (most similar) GT response.

4. EVALUATION ON BOGUS RESOLVERS

In the following, we present the findings of our analyses on open DNS resolvers. Our data is based on domain name scans and data aggregation between Jan 15 and Feb 10, 2015.

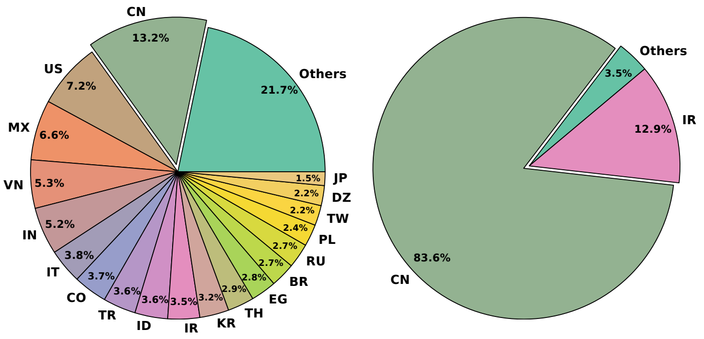
4.1 DNS-based Prefiltering

In the prefiltering process, we identified 85.8% (*MX* domain set) to 93.2% (*AV* set) of the aggregated DNS responses for the scanned domain sets (cf. Section 3.2) to be legitimate, considerably reducing the number of DNS responses we need to further analyze. Furthermore, 4.9–8.4% of the responses did not specify any IP addresses in the answer section. The highest value was observed for the *Malware* set, partially caused by DNS resolvers that aim to protect clients from accessing malicious domains. The number of *unexpected* ($domain \circ ip \circ resolver$) tuples ranges between 0.6% of all responses for the *MX* dataset and 4.4% for the *Malware* dataset, with a single exception for *NX* with 13.7% of unexpected responses. In total, we obtained 86,655,560 unexpected DNS responses (this does not include the empty DNS replies) from 19,180,169 distinct resolvers. Note that the high number of resolvers is caused by IP churn in combination with distributing our scanning activities over several days, as we often observed suspicious resolvers to switch to different IP addresses in the same ISPs' address spaces.

Up to 15.1% of the suspicious DNS resolvers for a particular domain set return their own IP for at least a single domain name. Moreover, 8,194 resolvers return their IP address for each requested domain in at least 75% of our domain sets. 5,404 of these IP addresses (65.9%) redirected to login web pages of routing equipment, distributed by two major manufacturers, while further 574 IPs (7.0%) are assigned to a specific brand of IP-based cameras. 50.4% of all suspicious DNS resolvers (i.e., 9,659,437 hosts) return the same set of IP addresses for more than one domain. In the extreme, 4.4% of all suspicious resolvers return even a single static IP address regardless of the domain name we queried. Further 2.0% return only `NS` records for the requested domain names, i.e., effectively denying recursive lookups.

4.2 HTTP Response Classification

When aggregating HTTP content for all unexpected DNS responses, we could obtain HTTP payload data for 88.9% of the ($domain \circ ip \circ resolver$) tuples. For the remaining 11.1% of tuples, we observe up to 65.1% of the suspicious DNS servers to return LAN IP addresses, while up to 32.2% replied with IP addresses located in the same AS or /24 network as the resolver. By taking a closer look at the rDNS records associated with these IP addresses we suspect some of the systems to belong to captive portals, which serve the actual login page to clients in specific IP ranges only. We verified other IP addresses to be associated with particular CDN providers (i.e., by checking AS and rDNS information). We assume these content servers to be disabled and not distributing actual HTTP(S) payload data, at least at the time of requesting content for our analysis. As such, certain resolvers might have delivered outdated IP address information for domain names associated with CDN providers.



(a) All responses (b) Unexpected responses

Figure 4: Resolver distribution per country for the domain names of Facebook, Twitter, and YouTube for (a) all DNS responses and (b) unexpected responses only

For non-legitimate tuples which provided HTTP data, we were able to classify 97.6–99.9% of the HTTP content. In fact, we classified over 99% of the responses for all but one domain category, indicating a close-to-exhaustive coverage of our analysis. Our clustering mechanism helped to group similar HTTP responses and served as data exploration step to further categorize the results. That is, we assigned labels to each individual cluster and mapped these labels to website categories that classify the type of served HTTP content. Table 5 summarizes these classification results. We mapped the response labels according to the following six categories:

Blocking. This category consists of websites to which a client system is redirected when the requested domain name has been blocked. This includes landing pages of multiple providers (e.g., parental control, ISPs or security organizations) when accessing forbidden or malicious content. We observe this type of data particularly for malware domains (e.g., 12,543 systems, i.e., 21.4% of the suspicious resolvers for the Virut domain `irc.zief.pl`, redirected to IP addresses to block the respective requests) and dating domains (e.g., 6,961 resolvers blocked the dating page `okcupid.com`, i.e., 10.9% of the suspicious DNS resolvers for this domain).

Censorship. We observe hundreds of thousands of suspicious resolvers to respond to requests of specific domains with a small set of IP addresses which we identified as landing pages for censorship. To distinguish between ordinary blocking and censorship, we closely analyzed the HTTP content and the behavior of the resolvers. If we find HTML content to specify text fragments such as `blocked by the order of [...] court/authority`, we flag the IP to be associated with censorship. We manually verified that the 299 IPs of landing pages are related to 34 different countries.

Next to the landing pages, we identified another type of censorship. When analyzing the legitimate and unexpected responses for particular domain names, we observe a conspicuous distribution of countries for the corresponding DNS resolvers. Figure 4-a illustrates that DNS resolvers are widely distributed in terms of geographical location for the combined set of DNS responses for Facebook, Twitter, and YouTube. When isolating the unexpected DNS responses, we find the majority of DNS servers to be located in China.

Table 5: Clustering and labeling results of the HTTP payload data for unexpected ($domain \circ ip \circ resolver$) tuples

	Average number of resolvers in % / (Highest number of resolvers seen for a domain in the particular dataset in %)													
Label	*Ads*	*Adult*	*Alexa*	*Antivirus*	*Banking*	*Dating*	*Filesharing*	*Gambling*	*GroundTr.*	*Malware*	*Misc.*	*MX*	*NX*	*Tracking*
Blocking	0.3	2.2	0.7	0.3	0.4	6.2	3.1	3.7	0.2	9.0	0.9	0.9	1.9	0.6
	(0.5)	(3.3)	(2.5)	(0.4)	(1.0)	(10.9)	(6.5)	(6.4)	(0.2)	(21.4)	(4.8)	(1.9)	(16.2)	(2.2)
Censorship	10.8	88.6	19.1	0.1	0.1	31.8	36.5	75.9	0.1	0.8	8.4	0.1	3.2	0.1
	(96.2)	(91.3)	(97.1)	(0.1)	(0.1)	(87.3)	(91.3)	(90.4)	(0.1)	(8.1)	(92.5)	(0.2)	(37.1)	(0.1)
HTTP Error	48.1	5.2	45.8	57.0	55.4	34.8	32.6	15.8	55.0	29.8	50.8	57.0	24.7	57.0
	(70.4)	(6.9)	(63.9)	(75.0)	(63.5)	(50.1)	(52.0)	(49.8)	(56.0)	(53.7)	(71.1)	(65.9)	(55.8)	(69.4)
Login	12.2	1.2	12.8	15.5	16.8	10.2	9.5	1.9	16.1	9.5	14.3	17.0	2.8	12.5
	(16.8)	(1.6)	(19.1)	(17.4)	(19.6)	(15.4)	(15.1)	(3.9)	(17.2)	(17.2)	(18.5)	(19.8)	(9.4)	(16.2)
Misc.	11.5	0.9	5.3	5.9	5.0	3.2	4.9	0.7	5.1	3.3	5.1	5.0	8.5	11.2
	(56.4)	(1.6)	(21.6)	(16.2)	(10.5)	(4.8)	(12.5)	(1.4)	(5.8)	(5.6)	(9.7)	(5.8)	(19.7)	(5.5)
Parking	17.1	1.8	16.1	21.2	22.2	13.8	13.4	2.0	23.4	26.2	20.5	20.0	23.2	18.6
	(23.9)	(2.4)	(24.0)	(25.0)	(24.3)	(21.5)	(22.4)	(2.4)	(23.9)	(92.1)	(83.6)	(23.4)	(42.4)	(24.0)
Search	0.0	0.1	0.2	0.0	0.1	0.0	0.0	0.0	0.1	21.4	0.0	0.0	35.7	0.0
	(0.1)	(0.1)	(2.7)	(0.1)	(0.1)	(0.1)	(0.0)	(0.0)	(0.6)	(69.3)	(0.5)	(0.1)	(65.1)	(0.0)

Interestingly, 2.4% of the Chinese resolvers (125,660 hosts) returned multiple DNS responses for particular domains: the first response specified a forged IP, while the second includes the legitimate IP address(es) for the requested domain, yet arriving a few milliseconds after receiving the bogus response. This effect is caused by the *Great Firewall of China* [3, 22, 42], which injects forged DNS responses upon monitoring DNS requests in the network traffic. Succeeding experiments underpin this assumption, as we obtain arbitrary IPs for specific domains when sending requests to randomly chosen IP ranges in Chinese networks, while requesting other domains does not trigger any response. We thus assume the *Great Firewall of China* to not modify responses of DNS resolvers but inject forged replies that (likely) arrive before the legitimate DNS responses at the client-side.

Figure 4-b outlines that 83.6% of the suspicious resolvers found for the domains Facebook, Twitter, and YouTube (i.e., 5,235,827 systems in total) are located in China, apparently returning randomly-chosen IP addresses. Further 805,559 systems (i.e., 12.9% of the suspicious resolvers for the three domains) are associated with Iranian censorship.

Besides China and Iran, we find more than 3 million resolvers that support censorship in other countries. Moreover, 491,032 resolvers (i.e., 74.2% of the hosts returning unexpected IP addresses for `adultfinder.com`) are associated with Indonesian censorship. Further 90.6% of the suspicious resolvers for domain *youporn.com* (696,777 systems in total) redirected to hosts that belong to Turkish (52.9%), Indonesian (29.3%), and Malaysian (8.4%) censorship. Yet, we also find Italian and Russian resolvers to be prevalent for *Filesharing* and *Gambling* domains. 92.1% of the suspicious resolvers for `rotten.com` (813,183 resolvers in total) are associated equally to Indonesian and Turkish censorship. For the domain `blogspot.com`, 396,681 resolvers (i.e., 88.5% of all suspicious DNS resolvers found for this domain name) redirected to IP addresses linked to Indonesian censorship.

Observing the prevalence of censorship in various countries, we wondered whether we observe all DNS resolvers located in a particular country to comply with censorship, indicating a strict policy or a government-controlled censorship system. We thus enumerated all resolvers that returned legitimate and unexpected responses by their geographical location and checked whether we find any DNS resolvers in a country to return legitimate responses, while others redirected to IP addresses assigned to censorship mechanisms.

For China, we indeed observe a high coverage: 99.7% of the Chinese resolvers returned bogus responses for Facebook, Twitter, and YouTube. For all other censoring countries, including Asia and Europe, we observe a lower coverage of blocking resolvers. 78.9% of the Mongolian resolvers blocked each of the four adult domains. For Indonesia, we observe varying numbers of resolvers blocking individual domains: a single adult domain is blocked by 91.6% of the Indonesian resolvers, while only 28.7% of the resolvers block another gambling and adult domain. In Europe, 83.9% of the Greek and 78.6% of the Belgian DNS resolvers restricted access to two gambling domains. Similarly, 69.3% of the Italian resolvers comply with censorship by blocking multiple online betting domains. Finally, 10.0% of the Turkish DNS resolvers did *not* censor otherwise-blocked domain names.

HTTP Error. This category includes web servers that either return HTTP error codes (`4xx`, `5xx`), respectively, error pages indicating invalid requests. Except for the domain sets *Adult* and *Gambling*—for which we find the majority of resolvers to return IP addresses assigned to censorship—an average of 40.6% of the suspicious resolvers per domain set directed to systems that serve various types of error pages, indicating that the requested content could not be found.

Login. About 10.9% of the suspicious DNS resolvers per domain set direct users to login websites, ranging from captive portals of ISPs, hotels, and educational institutions to router login pages and web mail logins. We observe 91.7% of the suspicious resolvers assigned to this category to forward client systems to login pages of routing equipment, manufactured by two large distributors of networking devices.

Parking. This category includes landing pages of parking providers, domain resellers, and hosting companies offering similar services. Parking can be observed most for the *Malware* domain set. 92.1% of the suspicious DNS resolvers for two of three Chinese domains in the *Malware* set (i.e., 1,371,67 systems) returned IP addresses that are linked to parking. It is known that domain resellers and parking providers tend to re-register malicious domain names to monetize the traffic [18, 20]. Yet, 261,116 systems (83.6% of the suspicious DNS resolvers for `torproject.org`) also redirect to parked content upon requesting this domain.

Search. The *Search* category groups all redirections to legitimate or mimicries of search pages. Responses like this are common for *NX* domain names—we find 35.7% of the suspicious DNS resolvers to redirect to search websites. Similar results were also observed for six out of 13 malware domains—even though these domains were existing when performing our scans. Similarly to parked domains, search websites might embed ad banners to monetize the traffic.

4.3 Miscellaneous Case Studies

Most of the HTTP content that was delivered by IP addresses in the non-legitimate DNS responses could be assigned to one of the categories outlined in Table 5. Yet, our in-depth analyses of HTTP(S), IMAP, POP3, and SMTP payload data also enabled us to classify content that was returned by a low number of suspicious resolvers. For reasons of brevity, we highlight the most interesting cases only.

Ad Redirects / Injections. We observe 281 suspicious DNS resolvers to return four IP addresses that redirected, respectively, replaced ad traffic of two large advertisement providers. Two of these hosts inject ad banners directly into the HTML content, while the other two IP addresses serve suspicious JavaScript code. We identify seven IP addresses from 14 suspicious resolvers that block ad traffic on purpose by replacing the ad banners with empty placeholders. Furthermore, two IP addresses were returned by seven DNS resolvers that serve content similar to the Google search web page, yet embedding ad banners underneath the search bar.

Transparent Proxies. We identify 20 IP addresses to act as proxies for all domain names, providing the same content as obtained from the original websites. We distinguish between proxies that support TLS encryption and provide the original certificate, and proxies that do not offer HTTPS. For proxies that forward valid certificate information for the requested domain names, we observe 99 DNS resolvers to return the associated 10 IP addresses. TLS prevents a potential eavesdropper from reading the actual content, yet traffic analysis may reveal insights to an adversary [11, 35]. In contrast, HTTP-only proxies might immediately eavesdrop on the communication channel, e.g., to sniff for credentials of login pages. We find 10,179 suspicious resolvers to return 10 IP addresses that proxy the original content, without accepting HTTPS requests. While we do not suspect all of these hosts to be malicious, users risk disclosing sensible login credentials when relying on these servers.

Phishing. Next to the proxies that serve original content for all requested domains, we also observe 39 hosts returned by 1,360 resolvers that provide content for particular domains only—potentially phishing for credentials. For PayPal, 176 resolvers served 16 IPs that provide content similar to the PayPal website. Yet, when investigating the HTML structure, we could reveal a phishing attempt: the body consists of 46 `<img>` tags reproducing the PayPal website and an HTML form that attempts to forward the entered login credentials to a `php` file via HTTP `POST`. Three of these servers accept HTTPS connections with self-signed certificates. Moreover, two suspicious phishing servers mimic an Italian banking domain. The first server is returned by 285 DNS resolvers and located in a Brazilian network. The second host is returned by 46 DNS resolvers and located in Russia. While the legitimate banking domain solely accepts HTTPS requests, none of the suspicious IP addresses does.

Mail Servers. We observe 64.7% of the suspicious DNS resolvers for the *MX* domain set to redirect to 1,135 IP addresses that listen to mail communication via IMAP, POP3, or SMTP. While we do not find any evidence of mail servers actively harvesting for login credentials (i.e., by attempting to log in with fake credentials and sending emails), attackers could secretly sniff on traffic. For Gmail and Yandex, we observe eight resolvers to point to IPs that provide similar or exactly the same banner information for SMTP as monitored for the legitimate SMTP servers. However, these bogus IPs are located in different networks (e.g., a Chinese research network), indicating that these might be suspicious.

Malware. Finally, 228 DNS resolvers redirect client systems to 30 distinct IP addresses that provide content similar to Adobe Flash and Java update pages, convincing the user to install potential software updates. We found these executables to be linked to malicious software. That is, we analyzed the samples dynamically and observed them to attempt downloading further (potentially malicious) executables. In line with our observations, the majority of anti-virus vendors classified these samples as malware downloaders.

5. DISCUSSION

We now discuss limitations of our methodology and conducted analyses, respectively, the DNS protocol in general.

Completeness. Performing Internet-wide DNS analyses implies several limitations on the significance of the obtained network data. First, DNS is a UDP-based protocol. As such, we might not obtain a response for each of our DNS requests due to packet loss. Second, a DNS packet might be corrupted, i.e., the checksum of the underlying UDP packet is invalid. As we do not know which part of the packet is bogus, we ignore invalid packets in all of our analyses. Also, our analyses are limited to *open* resolvers, while there is no reason to assume that closed resolvers do not likewise manipulate resolutions. Analyzing such resolvers, however, requires in-network measurement points such as Netalyzr [16].

Another concern regarding completeness is IP churn. DNS resolvers associated with dynamic broadband Internet links might be enumerated multiple times when they occasionally switch to IP addresses that are scanned in the later course of a DNS scan. Further, we might miss (parts of) the DNS responses in our domain scans when DNS resolvers become unavailable. Determining the error rate, however, is unfeasible as resolvers might simply ignore particular requests, as observed for censorship or badly configured resolvers. Hence, we cannot distinguish between requests that are unanswered on purpose or due to the above-mentioned limitations. Furthermore, the number of identified resolvers fluctuates per day and time we perform our scans. As such, when initiating multiple IPv4 scans on the same day, the number of active resolvers might differ due to the various geographical locations and time zones. To cope with these limitations, we adjusted the rate of outgoing DNS requests to achieve a low packet loss and started our weekly scans at the same time (on Friday evenings CET). To reduce the impact of IP churn, we scanned the domains in a timely manner, i.e., the delay between an Internet-wide scan and the aggregation of HTTP data (and potentially requesting further domains at these resolvers upon redirections) was at most 8 hours.

In order to scale, our domain set is limited to 155 domain names of 13 different categories that could be relevant for DNS response forgery. As such, our dataset is far from complete and we consequently miss suspicious activities for domains that are not covered by the set. Performing analyses on dozens or hundreds of additional domains, however, would significantly increase the impact on the individual resolvers as well as the amount of data that needs to be processed in a timely manner. Furthermore, the dataset would still remain incomplete, as attackers may choose arbitrary domain names to perform suspicious or malicious activities.

Our evaluation results are constrained to suspicious DNS activities performed in the period from Jan 15 to Feb 10, 2015, thus rather represent a snapshot of the DNS threat landscape. Repeated analyses could have been performed, however, we refrained from doing so for multiple reasons. First, as we aimed to obtain a detailed understanding of all suspicious DNS resolvers in IPv4 at once, we performed one large evaluation instead of conducting (repeated) smaller analyses on subsets of open resolvers, which would increase the impact on these servers. Second, we assume that repeated analyses will only provide minor changes in the DNS threat landscape, thus do not justify repeated scanning.

Finally, we used heuristics and clustering to group DNS and HTTP responses. This way, we could label about 99% of the responses, leaving a minor percentage unclassified. We argue that the remaining responses represent rather uninteresting behavior, as attackers typically strive for larger coverage (thus would have been identified in our analyses). We still manually investigated a random sample set of the unclassified HTTP responses and found them to be associated with personal, shopping, or similar categories of websites.

Illegitimacy. Multiple IP addresses acted as an HTTP(S) proxy between a client system and a web server of the requested domain name. As such, we obtained similar or the same HTML content as provided by the legitimate IP address(es). By analyzing the HTTP payload of websites only, we are limited to identifying IP addresses that could potentially be performing malicious activities. That is, we have no evidence of eavesdropping activities, yet attackers could trivially sniff on traffic (especially for HTTP-only proxies).

We queried dozens of domain names at millions of open DNS resolvers, causing network traffic which might unintentionally harm a remote network. Within our long-term empirical measurement study of DNS resolvers, however, we have not received a single abuse complaint that our Internet-wide scanning activities caused a congestion of either an open resolver or an AuthNS of a scanned domain. As such, we assume the combination of rate-limiting the outgoing DNS requests and distributing the sequence of target IP addresses over time using an LFSR to be reasonable in practice.

In addition, one has to be careful judging certain politically-driven actions (such as censorship) as "malicious". Tolerating the constitutions and cultures of other countries, we merely provide our rational insights and technical inputs.

DNS Authenticity. The DNS protocol—in general—does not deploy any security measures to validate the data authenticity of DNS responses, hence security extensions such as DNSSEC [5] have been proposed and implemented. It yet has to be evaluated whether the DNS extensions also protect against strong "adversaries" such as the Chinese firewall, which inject forged DNS responses that arrive way ahead of the legitimate replies at the requesting clients. As a resolver typically utilizes the first response that matches an open transaction (and ignores succeeding responses), DNSSEC does not effectively protect against this kind of attack unless the client *waits* for a correctly signed DNS response and drops all previously incoming unsigned and incorrectly signed replies [5, 6, 43]. Yet, this strategy can be deployed only when either (i) all domains globally deploy DNSSEC, or (ii) the client gained previous knowledge that the particular domain indeed supports DNSSEC, thus only a signed DNS response is acceptable at all. As of May 2015, the global coverage of DNSSEC was rather low [21, 37, 38], e.g., less than 0.6% of the domains associated with the `.net` TLD operated DNSSEC. Further, there is no reliable way for a client to determine the state of the DNSSEC deployment for specific domains (except by performing DNS requests and check the replies—yet these could also be forged). As such, the DNS protocol and its security extensions may require improvements to cope with attacks or censorship measures such as packet injections conducted by the *Great Firewall of China*.

6. RELATED WORK

This work is inspired by the analysis of corrupted DNS resolution paths by Dagon et al. [10] in 2008. The authors analyzed DNS resolutions of 600,000 open resolvers for 84 domain names to identify suspicious responses. Similar to our work, they requested `A` records for various banking, antivirus, and search engine websites. The aggregated responses, however, were analyzed and presented rather coarse-grained, i.e., the authors performed a manual analysis on 250 randomly sampled web pages only and did not provide detailed statistics about their findings—except illustrating a limited number of categories such as Chinese splash pages. Furthermore, it remains rather unclear how the authors identified the legitimate IP addresses of each domain, particularly important when filtering IP addresses of CDNs that may span hundreds of ASes. In our work, we outline our prefiltering and identification of legitimate IP addresses in detail and perform a more fine-granular and thorough analysis of the unexpected DNS responses to cluster and classify non-legitimate DNS responses accordingly, resulting in a more comprehensive overview of the DNS resolver behavior. As such, we are the first to perform an in-depth analysis of incorrect DNS resolutions provided by all open DNS resolvers operated in the entire IPv4 address space. That is, we were capable of labeling about 99% of the obtained DNS and HTTP responses, achieving fine-granular classification results on the DNS resolution paths of open DNS resolvers.

Prior work by Weaver et al. [40, 41] analyzed DNS traffic aggregated from Netalyzr sessions, a Java-based applet running on individual client systems of volunteers. The authors mostly focused on DNS error monetization—a technique to redirect clients to ad websites upon requests for non-existent domains. Yet, the authors also observed sessions, in which DNS resolvers redirect clients to websites of malicious character or deliberately disabled the resolution of the Windows update website to prevent system updates. Unfortunately, it remains unclear how the authors identified the corresponding DNS servers to block the update page on purpose and not due to erroneous configurations. The results further provide rather little insight, as the analyses were conducted on limited DNS traffic, and as such the number of analyzed DNS resolvers was low. Contrary, we perform a large-scale analysis on all resolvers in the entire IPv4, achieving a more detailed insight to the resolvers' landscape. Still, a benefit of the approach proposed by Weaver et al. is the detection of suspicious and malicious DNS resolutions caused by closed DNS resolvers, while we have to stick to the analysis of open DNS resolvers only. Combining both approaches thus presumably increases the detection of forged DNS resolutions.

We group further related works by their topic.

Measurements on the DNS Protocol. A large body of work exists on analyzing DNS resolvers [1, 4, 25, 33]. Most of these analyses are conducted on a small subset of all resolvers. As such, it is unclear if the observed results generalize to all resolvers world-wide. Sisson [34] analyzes open resolvers based on sampled scans that repeatedly query the same set of resolvers, thus covering only a small fraction of all open resolvers. We omit a detailed comparison, as Sisson scans only known AuthNS and resolvers contacting their AuthNS, therefore relying on a non-random sample. Overall, their measurements cover less than 0.2% of the resolvers in our dataset. Jiang et al. [15] analyzed the caching behavior of resolvers. The authors identified an attack vector in DNS software that allows to extend the caching of domains even after they have been removed from the upper DNS hierarchy. Schomp et al. [31] randomly probed the IPv4 address space to enumerate DNS resolvers and distinguish between recursive DNS resolvers and DNS proxies. Furthermore, the authors closely analyzed the caching behavior of resolvers in more detail. Similar to our work, the authors performed device fingerprinting of the identified DNS resolvers, yet without providing an overview of the results. The authors further focused on device information provided by the HTTP header only, thus missing various device categories, e.g., provided by the HTTP body, respectively, other protocols such as FTP and Telnet. Takano et al. [36] performed measurements based on responses for Internet-wide CHAOS `version.bind` requests. The authors primarily focused on DNS server software and their distribution in each *Regional Internet Registry*. Contrary to our work, no device fingerprinting was conducted. Concurrent to our work, Scott et al. [32] probed the IPv4 address space for open resolvers to analyze DNS resolutions. Instead of focusing on suspicious DNS resolution paths, the authors queried the Top 10,000 Alexa domain names at the identified resolvers to analyze the infrastructure of *Content Delivery Networks*. By deploying automated clustering mechanisms, they were capable of accurately detecting CDN deployments in their scanning results. Contrary, we performed the identification of CDNs by leveraging AS, rDNS, and HTTPS certificate information in our prefiltering step. Combining both techniques could improve the detection capabilities of CDNs and thus the effectiveness of our prefiltering, reducing the amount of unfiltered but legitimate DNS responses that needs to be processed in the latter processing steps.

Censorship. Bailey and Labovitz [7] provided an overview of various countries restricting access to certain websites. Verkamp and Gupta [39] focused on how censorship is technically implemented. By using PlanetLab nodes and personal contacts in various countries, the authors identified all eleven examined countries to perform censorship, either by filtering at the DNS-level or directly on URLs and keywords. In contrast, we obtained responses directly from arbitrary DNS resolvers world-wide. As such, our results are not derived from a limited number of hosts per country. Levis [19] focused on censorship conducted by injecting forged DNS packets into network traffic, e.g., performed by the *Great Firewall of China* [3, 22, 42]. The author claims that Chinese censorship may not only affect clients in China but also systems in other countries when their DNS traffic is routed through China via transit—an observation we can confirm for our datasets. Apart from China, similar behavior is also observed for other countries, e.g., 56.9% of the Estonian DNS resolvers respond with IP addresses for gambling domains that we assigned to Russian censorship.

Internet-wide Scanning. Durumeric et al. [12] proposed ZMap, a high-speed application to run Internet-wide scans. While we do not leverage ZMap for our scans, we apply many of the guidelines and techniques. Rossow [29] and Kührer et al. [17] performed Internet-wide scans for multiple UDP- and TCP-based protocols to identify and monitor systems that are prone to abuse for amplification DDoS attacks.

7. CONCLUSION

In this paper, we studied the landscape of DNS resolvers on long term, i.e., we analyzed empirical data of Internet-wide DNS scans we performed for more than one year. More precisely, we analyzed the fluctuation of DNS resolvers over time and classified the resolvers according to the operated DNS server software, underlying hardware specifications, and their utilization by actual client systems. To analyze further attack vectors in the *Domain Name System*, we took the viewpoint of a client system and determined the response authenticity and integrity of all open DNS resolvers in the entire IPv4 address space. By performing billions of DNS lookup requests for 155 domains at millions of open resolvers, we identified millions of resolvers that deliberately manipulated DNS resolutions and returned unexpected IP address information. Besides legitimate redirections (e.g., to captive portals such as router login pages), our analyses revealed thousands of resolvers that manipulated DNS resolutions to censor communication channels, inject advertisements, serve malicious files, or perform phishing. As such, the DNS protocol does not only have flaws at the network-level in terms of traffic amplification vulnerabilities [17, 24, 29] but also lack verification mechanisms at the application-level to sufficiently protect end hosts from malicious resolvers that redirect clients to suspicious content.

Acknowledgment

This work was supported by the German Federal Ministry of Education and Research (BMBF Grant 01IS14009B BDSec). We would like to thank our shepherd Georgios Smaragdakis for his support in finalizing this paper and the anonymous reviewers for their constructive and valuable comments.

8. REFERENCES

[1] Ager, B., Mühlbauer, W., Smaragdakis, G., and Uhlig, S. Comparing DNS Resolvers in the Wild. In *Internet Measurement Conference* (2010).

[2] Alexa Internet, Inc. Top 1 Million Traffic Ranking. `http://www.alexa.com/topsites/`.

[3] Anonymous. Towards a Comprehensive Picture of the Great Firewall's DNS Censorship. In *USENIX Workshop on Free and Open Communications on the Internet* (2014).

[4] Antonakakis, M., Dagon, D., Luo, X., Perdisci, R., Lee, W., and Bellmor, J. A Centralized Monitoring Infrastructure for Improving DNS Security. In *Symposium on Recent Advances in Intrusion Detection* (2010).

[5] Arends, R., Austein, R., Larson, M., Massey, D., and Rose, S. RFC 4033: DNS Security Introduction and Requirements, 2005.

[6] Atkins, D., and Austein, R. RFC 3833: Threat Analysis of the Domain Name System (DNS), 2004.

[7] Bailey, M., and Labovitz, C. Censorship and Co-option of the Internet Infrastructure. *Ann Arbor 1001* (2011).

[8] Blackford, J., and Digdon, M. TR-069, CPE WAN Management Protocol, CWMP Version: 1.4. `http://www.broadband-forum.org/technical/download/TR-069_Amendment-5.pdf`, 2013.

[9] Dagon, D., Antonakakis, M., Vixie, P., Jinmei, T., and Lee, W. Increased DNS Forgery Resistance Through 0x20-Bit Encoding: SecURItY viA LeET QueRieS. In *ACM Conference on Computer and Communications Security* (2008).

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

[11] Danezis, G. Traffic Analysis of the HTTP Protocol over TLS. `http://www0.cs.ucl.ac.uk/staff/G.Danezis/papers/TLSanon.pdf`, 2009.

[12] Durumeric, Z., Wustrow, E., and Halderman, J. A. ZMap: Fast Internet-Wide Scanning and its Security Applications. In *USENIX Security Symposium* (2013).

[13] Eastlake, D. RFC 5395: Domain Name System (DNS) IANA Considerations, 2008.

[14] Grangeia, L. DNS Cache Snooping. Tech. rep., Securi Team-Beyond Security, 2004.

[15] Jiang, J., Liang, J., Li, K., Li, J., Duan, H., and Wu, J. Ghost Domain Names Revoked Yet Still Resolvable. In *Symposium on Network and Distributed System Security* (2012).

[16] Kreibich, C., Weaver, N., Nechaev, B., and Paxson, V. Netalyzr: Illuminating The Edge Network. In *Internet Measurement Conference* (2010).

[17] Kührer, M., Hupperich, T., Rossow, C., and Holz, T. Exit from Hell? Reducing the Impact of Amplification DDoS Attacks. In *USENIX Security Symposium* (2014).

[18] Kührer, M., Rossow, C., and Holz, T. Paint It Black: Evaluating the Effectiveness of Malware Blacklists. In *Symposium on Recent Advances in Intrusion Detection* (2014).

[19] Levis, P. The Collateral Damage of Internet Censorship by DNS Injection. *ACM SIGCOMM CCR 42*, 3 (2012).

[20] Li, Z., Alrwais, S., Xie, Y., Yu, F., and Wang, X. Finding the Linchpins of the Dark Web: A Study on Topologically Dedicated Hosts on Malicious Web Infrastructures. In *IEEE Symposium on Security and Privacy* (2013).

[21] Lian, W., Rescorla, E., Shacham, H., and Savage, S. Measuring the Practical Impact of DNSSEC Deployment. In *USENIX Security Symposium* (2013).

[22] Lowe, G., Winters, P., and Marcus, M. L. The Great DNS Wall of China. *MS, New York University 21* (2007).

[23] M. Prince; CloudFlare, Inc. `http://blog.cloudflare.com/the-ddos-that-almost-broke-the-internet`, March 2013.

[24] MacFarland, D. C., Shue, C. A., and Kalafut, A. J. Characterizing Optimal DNS Amplification Attacks and Effective Mitigation. In *Passive and Active Measurement Conference* (2015).

[25] Mao, Z. M., Cranor, C. D., Douglis, F., Rabinovich, M., Spatscheck, O., and Wang, J. A Precise and Efficient Evaluation of the Proximity Between Web Clients and Their Local DNS Servers. In *USENIX Annual Technical Conference* (2002).

[26] MaxMind GeoIP Database. `http://www.maxmind.com/en/ip-location`.

[27] Open Resolver Project. `http://openresolverproject.org/`.

[28] Rajab, M. A., Monrose, F., Terzis, A., and Provos, N. Peeking Through the Cloud: DNS-Based Estimation and Its Applications. In *Applied Cryptography and Network Security* (2008).

[29] Rossow, C. Amplification Hell: Revisiting Network Protocols for DDoS Abuse. In *Symposium on Network and Distributed System Security* (2014).

[30] Rossow, C., Dietrich, C. J., Bos, H., Cavallaro, L., van Steen, M., Freiling, F. C., and Pohlmann, N. Sandnet: Network Traffic Analysis of Malicious Software. In *Workshop on Building Analysis Datasets and Gathering Experience Returns for Security* (2011).

[31] Schomp, K., Callahan, T., Rabinovich, M., and Allman, M. On Measuring the Client-Side DNS Infrastructure. In *Internet Measurement Conference* (2013).

[32] Scott, W., Berg, S., and Krishnamurth, A. Satellite: Observations of the Internet's Star. Tech. rep., University of Washington, 2015.

[33] Shue, C. A., and Kalafut, A. J. Resolvers Revealed: Characterizing DNS Resolvers and their Clients. *ACM Transactions on Internet Technology 12*, 4 (2013), 14.

[34] Sisson, G. DNS Survey: October 2010. Tech. rep., The Measurement Factory, 2010.

[35] Sun, Q., Simon, D. R., Wang, Y.-M., Russell, W., Padmanabhan, V. N., and Qiu, L. Statistical Identification of Encrypted Web Browsing Traffic. In *IEEE Symposium on Security and Privacy* (2002).

[36] Takano, Y., Ando, R., Takahashi, T., Uda, S., and Inoue, T. A Measurement Study of Open Resolvers and DNS Server Version. In *Internet Conference (IEICE)* (2013).

[37] van Rijswijk-Deij, R., Sperotto, A., and Pras, A. DNSSEC and Its Potential for DDoS Attacks: A Comprehensive Measurement Study. In *Internet Measurement Conference* (2014).

[38] VeriSign, Inc. DNSSEC Scoreboard. `http://scoreboard.verisignlabs.com/`.

[39] Verkamp, J.-P., and Gupta, M. Inferring Mechanics of Web Censorship Around the World. *Free and Open Communications on the Internet* (2012).

[40] Weaver, N., Kreibich, C., Nechaev, B., and Paxson, V. Implications of Netalyzr's DNS Measurements. In *Workshop on Securing and Trusting Internet Names* (2011).

[41] Weaver, N., Kreibich, C., and Paxson, V. Redirecting DNS for Ads and Profit. In *USENIX Workshop on Free and Open Communications on the Internet* (2011).

[42] Xu, X., Mao, Z. M., and Halderman, J. A. Internet Censorship in China: Where Does the Filtering Occur? In *Passive and Active Measurement Conference* (2011).

[43] Zmijewski, E. DNS: When Governments Lie. `http://research.dyn.com/2010/12/dns-when-governments-lie-2/`, December 2010.