The Collateral Damage of Internet Censorship by DNS Injection *

Sparks
Hovership Nebuchadnezzar
Zion Virtual Labs
zion.vlab@gmail.com

Neo†
Hovership Nebuchadnezzar
Zion Virtual Labs
zion.vlab@gmail.com

Tank
Hovership Nebuchadnezzar
Zion Virtual Labs
zion.vlab@gmail.com

Smith
Hovership Nebuchadnezzar
Zion Virtual Labs
zion.vlab@gmail.com

Dozer
Hovership Nebuchadnezzar
Zion Virtual Labs
zion.vlab@gmail.com

ABSTRACT

Some ISPs and governments (most notably the Great Firewall of China) use DNS injection to block access to "unwanted" websites. The censorship tools inspect DNS queries near the ISP's boundary routers for sensitive domain keywords and injecting forged DNS responses, blocking the users from accessing censored sites, such as `twitter.com` and `facebook.com`. Unfortunately this causes large scale collateral damage, affecting communication beyond the censored networks when outside DNS traffic traverses censored links. In this paper, we analyze the causes of the collateral damages comprehensively and measure the Internet to identify the injecting activities and their effect. We find 39 ASes in China injecting forged replies even for transit DNS traffic, and 26% of 43,000 measured open resolvers outside China, distributed in 109 countries, may suffer some collateral damage. Different from previous work, we find that most collateral damage arises from resolvers querying TLD name servers who's transit passes through China rather than effects due to root servers (F, I, J) located in China.

Categories and Subject Descriptors

C.2.0 [**Computer Communication Networks**]: General

General Terms

Measurement, Security

Keywords

DNS, packet injection, Internet measurement, Internet censorship, Great Firewall of China, collateral damage

1. INTRODUCTION

Since DNS is essential for effectively all communication, it is a common target for censorship systems. The most popular approach involves packet injection: a censorship system observes DNS requests and injects fake replies to block communication. Yet censorship systems may affect more than just the censored network.

As a concrete example, consider a query for `www.epochtimes.de` from a US user, using a US-based DNS resolver. The US resolver will need to contact one of the DNS TLD authorities for `.de`, located in Germany. If the path to the selected TLD authority passes through China, then the Chinese Great Firewall will see this query and inject a reply which the US resolver will accept, cache, and return to the user, preventing the user from contacting the proper web server.

Packet injection's popularity as a censorship mechanism arises from its ease of implementation. The censor needs to only monitor traffic and inject responses. Thus network operators have used TCP packet injection to block Peer to Peer traffic [4] or undesirable web content [3], and the Chinese Great Firewall and others use DNS packet injection to block entire sites. While some ISPs are content to block users inside their network from accessing "unwanted" websites using DNS injection, they may not know that their DNS injecting activities potentially affect users outside their network. In the motivating example of contacting `www.epochtimes.de` from the US, the *collateral damage* was due solely to the DNS request passing through a censored network as traceroute verified that the path for HTTP traffic did not pass through a censored network.

Although the DNS community has perceived such collateral damage, they only found it happened when resolvers outside contacted DNS authorities inside the censored country [1], with the most famous examples involving queries from Chile that found themselves routed to the Chinese I-root server [6].

However, the range of the potential damage is actually much more complicated. We find that even querying name servers unrelated to censored countries, resolvers outside could still suffer from collateral damage caused by DNS injection activities from censored transit networks.

In this paper, we make a comprehensive study of the collateral damage caused by DNS injection. Specifically, we try to answer the following three questions:

- How does this collateral damage occur?
- Which ISPs are adopting DNS injection?
- What names and resolvers are affected?

*We use pseudonyms to protect the authors.

†Corresponding author.

For the first question, we analyze the cause from the diversity of DNS resolution paths, as well as the dynamic routing. We utilize two tools, *HoneyQueries* to detect affected paths and *TraceQueries* to detect the point of injection. This enables us to identify the censored ASes. Finally, we perform measurements using *StepNXQueries* which allow us to detect whether a resolver's path to the authorities for the root or a given TLD experience censorship. A survey of 43,842 non-censored resolvers showed 11,579 suffering from some collateral damage. Unlike the results in [1], we find that the most common source of pollution exists on the path between the resolvers and the TLD authorities, particularly the paths to `.de` and `.kr` authorities.

The rest of the paper is organized as follows. In § 2 we give a brief introduction to DNS resolution and how packet injection can disrupt the process. Then we analyze the cause for the collateral damage caused by DNS injection in § 3. In § 4 we describe our experiment methodologies and present the experiment results. We have a discussion in § 5 before concluding in § 6.

2. BACKGROUND

The standard DNS resolution process [8, 9, 5] consists of several pieces, including the stub resolver on the user's computer, the recursive resolver, the root servers ("."), Top Level Domain (TLD) authorities, and the site's authority nameservers as illustrated in Figure 1. When a user generates a request to the recursive resolver, and the resolver has no valid cache information, it first directs that question in full to a root server, which redirects the resolver to the TLD authorities, which redirect to the final authority servers. In the process the resolver caches the intermediate information as well as the final answer.

If an attacker, be it a hacker, an ISP, or a government, can monitor any of the links and inject packets, he can launch a DNS injection attack, replying with a forged response which has the appropriate query question and protocol identifiers but with a bogus DNS answer, mapping the queried domain to either an invalid IP address or an IP address controlled by himself. In the absence of DNSSEC validation, the resolver will generally accept the faked answer because it arrives earlier than the real one, and, as a result, the access to the sensitive site will be blocked or redirected.

The ease of this attack makes it naturally an effective censorship mechanism. It is well known that the Great Firewall of China (GFC) uses this mechanism. The survey of [7], in which the authors queried $>$ 800 DNS resolvers in China, found that 99.88% of them were affected by the GFC.

The collateral damage of GFC was first discussed in a DNS operation mailing list when a Chilean operator found that queries from Chile and California to I.RootServer.NET sometimes experienced DNS pollution [6]. In [1], Brown et. al. analyzed this incident and determined that this kind of pollution could affect many countries because three root DNS server nodes (F, I, and J) have anycast instances in China. They believed that after Netnod withdrew the anycast routes for the Chinese I-root nameserver from CNNIC, the collateral damage should disappear.

Yet there exists an additional collateral damage mechanism. Resolvers only rarely query the DNS root as the root's responses are broad and long lived, lasting in the cache. Yet resolvers must frequently query the TLD authorities. Thus the paths from the resolver to the TLD authorities is as critical as the path from the resolver to the roots.

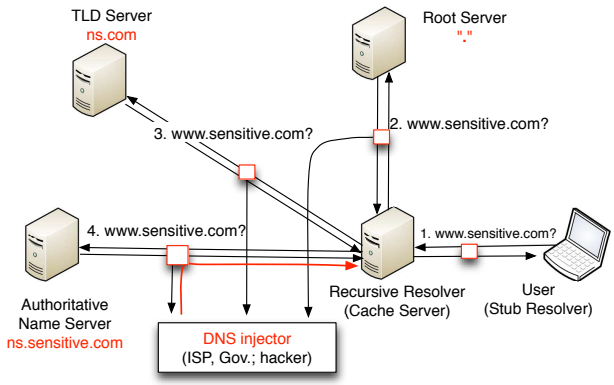


Figure 1: DNS query process and DNS injection

3. CAUSES OF COLLATERAL DAMAGE

Collateral damage occurs when a DNS query from a recursive resolver enters a censored network, causing the censorship mechanism to react. Although intuition would suggest that this would be a rare occurrence, there exist several factors which may cause the censor to receive and react to DNS queries from outsiders.

Iterative Queries: A recursive resolver does not send limited queries, such as asking the root for just the nameservers of the desired TLD. Instead, if it lacks cache entries for the TLD authorities, it sends the *entire* query to a root server. Similarly, the resolver sends the entire query to a TLD authority if there are no cache entries for the domain's authority.

This may be further complicated by "out-of-bailiwick" glue records. Suppose the DNS authorities for `example.com` are `ns1.example.net` and `ns2.example.net`. In the absence of cached data, the resolver will first query for `www.example.com` to a root server and then to a `.com` TLD authority. The reply from the `.com` TLD will now cause the resolver to first query for `ns1.example.net` before resuming the query for `www.example.com`. Thus the resolver will query for `www.example.com` three times: to a root, to a `.com` TLD server, and to `ns1.example.net`, and at least two queries for `ns1.example.net`: to a root and to a `.net` TLD server[1]. Thus a simple "lookup" may generate numerous queries, the disruption of any by censorship would cause resolution to fail.

Redundant Servers and Anycast: Most DNS deployments use multiple servers in multiple networks to increase reliability [2], and actual selection of particular authorities by a recursive resolver is a complex topic, with nameservers using various algorithms. Thus, with 13 different roots and 13 global TLD servers for `.com`, a resolver may experience collateral damage if a path to any one of these 26 IPs passes into a censored network.

Further complicating the picture is the use of *anycast* [10] DNS authorities, where a single IP address may represent a widely deployed system of servers. Two resolvers in different networks may reach different physical servers, along very

[1]If the authority for `example.net` is `ns1.example.net`. Otherwise it can generate even more requests

different paths, even though they are attempting to contact the same IP address.

Censored Transit and Dynamic Routing: The paths from the resolver to the authorities is dynamic, routing through a series of Autonomous Systems (AS), independent networks which together form the Internet. If one transit AS implements censorship, then all traffic which passes through that AS experiences censorship, even if both the source and destination are in non-censored networks. Routing changes also make it difficult to predict when and where DNS queries will pass through censored transit networks.

4. MEASUREMENT AND RESULTS

By measuring of the effect of DNS injection, we want to answer the following two questions related to the collateral damage:

(1) How many ISPs and ASes implement DNS injection-based censorship?

(2) How widely are DNS resolvers suffering from collateral damage due to censorship, and what is the cause of this collateral damage?

4.1 Searching for Injected Paths: Honey-Query

In order to measure the impact of injection on users outside the censored networks, we must first identify and exclude the networks which use DNS injection for censorship. Based on our previous experience with censored networks and the work of Lowe et al[7], we make two assumptions: the DNS injection occurs in the core or on the border of the networks and the DNS injector does not consider packet origin when injecting packets. If the censorship occurs in the edge connecting the user it is highly unlikely to cause collateral damage, and a censor which considers packet origin would not cause collateral damage.

Like the concept of a *Honeytoken* [11], we launch a large amount of *HoneyQueries* to search for the injected paths. These queries target non-responsive IPs with queries to a sensitive domain name. Because the query only targets non-nameservers, any DNS response is likely due to packet injection.

Probing Targets: In order to search all possible AS-level paths, ideally we should make sure that our HoneyQuery probing covers all ASes in the Internet. We select an IP address in each /24 of the IPv4 address and verify that the IPs are not running DNS servers. We then probe these 14 million target IPs with our HoneyQueries.

Vantage Point: Other observers [6, 7] and our own experience show that these injectors fake answers for both inbound and outbound DNS queries. Therefore, our HoneyQuery probing could possibly cover all ASes from a single vantage point as long as its not in a censored network. There does exist a minor false-positive: if an uncensored network receives transit from a censored network from our vantage point but not for other traffic. We are unable to determine when this occurs, and simply treat such networks as censored for later analysis. We selected a virtual private server (VPS) in AS 40676 (Psychz Networks) in US as our vantage point.

Domain Names For Testing: Experientially, we select 10 domain names for the probing(Table 1), including some social networks, pornography, web hosting, blogs, stream media, and search engines which we would expect to be targets of government or ISP censorship.

Domain Name	Category
`www.google.com`	Search Engines
`www.facebook.com`	Social Networks
`www.twitter.com`	Social Networks
`www.youbube.com`	Streaming Media
`www.yahoo.com`	News Portal
`www.appspot.com`	Web Hosting
`www.xxx.com`	Pornography
`www.urltrends.com`	Sites Ranking
`www.live.com`	Portal
`www.wikipedia.org`	Reference

Table 1: Domain Names for Probing.

Region	IP Count	Percentage
CN	388206	99.80
CA	363	0.09
US	127	0.03
HK	111	0.03
IN	94	0.02
Total 16 regions		

(a) Top 5 regions.

AS number	Region	IP Count	Percentage
4134	CN	140232	36.05
4837	CN	88573	22.77
4538	CN	35217	9.05
9394	CN	24880	6.40
4812	CN	14913	3.83
Total 197 ASes			

(b) Top 5 ASes.

Table 2: Statistics of the `Poisoned_IP_List` collected from HoneyQuery probing.

HoneyQuery Probing: We send HoneyQueries with domain names above to all the target IP addresses from the vantage point. If there is any response for a HoneyQuery, we mark the domain name as blacklisted and the target IP as a poisoned IP. We also collect all the IPs used in the injected responses(we call them lemon IPs). After HoneyQuery probing, we get three lists: (1)`Blacklisted_Domain_List`, containing poisoned domain names in testing domain name set; (2)`Poisoned_IP_List`, containing IPs suffering directly from censorship; (3)`Lemon_IP_List`, containing the IPs used in all the bogus responses (allowing us to recognize consistently censored results).

We conducted our HoneyQuery probing during November, 2001 and obtained a poisoned IP list of 388,988 IP addresses, distributed in 16 regions (CN, CA, US, HK, IN, AP, KR, JP, TW, DE, PK, AU, SG, ZA, SE, FI) and 197 ASes. The top regions and ASes are shown in Table 2.

For the IPs in the `Poisoned_IP_List`, its location (region or AS) does not mean that the hosting AS or region injects the faked DNS response; but means there should be an injector on the transit path from our vantage point. We will locate the injectors in § 4.2.

We obtained six domain names in the `Blacklisted_Domain_List`: `www.facebook.com, twitter.com, www.youtube.com, www.appspot.com, www.xxx.com, www.urltrends.com`, and 28 different IPs in the `Lemon_IP_List`, allowing us to easily

Region	Count	Percentage
US	12519	28.76
JP	4889	11.23
RU	3306	7.60
DE	2345	5.39
TW	1733	3.98
GB	1580	3.63
CA	1150	2.64
IT	1053	2.42
Total 173 regions		

Table 4: Distribution of open resolvers for StepNXQuery probing.

create queries that may experience censorship and a list of known-bad results

4.2 Locating Injecting ISPs: TraceQuery

Given the list of censored IPs, we now identify the network location of the injectors using a *TraceQuery*.

A TraceQuery is a crafted DNS query with a domain name in `Blacklisted_Domain_List` and a customized TTL in IP header. Like *traceroute*, TraceQuery utilizes TTL decrements to ensure that the packets expire in the network. When the query goes through the network, each router along the path will decrease the IP TTL by one. Once the IP TTL gets zero, the router will drop the packet, and send back an ICMP time exceed message, allowing us to record the network path. The queries which pass an injector also trigger a DNS reply before expiring.

By conducting a TraceQuery to the final destination in the Poisoned IP list, this reveals all the DNS injectors in the path and their locations in the network.

After TraceQuery probing, we obtained a list of 3,120 router IPs associated with DNS packet injection, belonging to only 39 Chinese ASes. Table 3 shows the information of top ten poisoning ASes. Thus we conclude that the non-Chinese IPs in our poisoned IP list are due to either errors in geolocation or Chinese transit for non-Chinese traffic.

4.3 Evaluating the Collateral Damage: StepNXQuery

Given the list of ASes that inject DNS replies, the question remains: does the censorship imposed within these ASes affect external resolvers? We probe for such collateral damage using a list of 43,842 non-censored open recursive resolvers distributed in 173 countries (Table 4).

We probe these resolvers from our non-poisoned vantage-point with names derived from the Blacklisted Domain List, comparing with the replies in the Lemon IP list to see if the resolver is generally poisoned. We conduct these probes using TCP, to further reduce the likelihood that the communication with the resolver encounters censorship.

Yet simple poisoning is not the only concern: if there exists a censored path from the resolver to the root, or from the resolver to a TLD authority, that path may also poison results. Thus we develop and utilize a series of *S*tepNXQueries. We structure these queries to take advantage of over-eager pattern matching in the censorship systems, which regard names such as `www.facebook.com.fu` as objectionable.

Thus we can guarantee that a query from the recursive resolver goes to a specific level in the DNS hierarchy by generating an NXNAME (No Such Name) triggering request. Thus, to test the root path from the resolver, we query for names like `www.facebook.com.{RANDOM}`, with `RANDOM` being a random string which will generate an NXNAME response from the root. By repeating this test 200 times with different random strings, we take advantage of the recursive resolver's willingness to distribute queries between authorities to test all paths to the root servers from the given resolver.

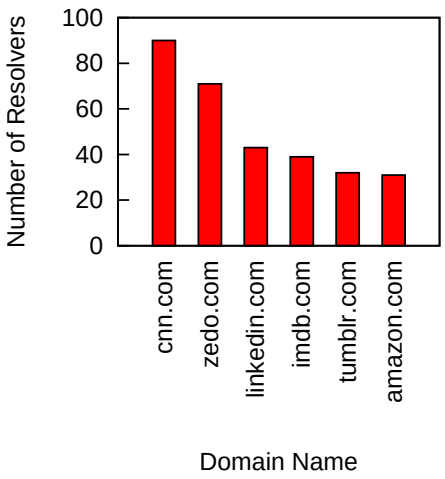


Figure 4: Affected domain names.

The same technique allows us to probe the path between the resolvers and the TLD servers, replacing {`RANDOM`} with {`RANDOM`}`.tld`. Since the TLD information is already cached with a long TTL, these queries only traverse the path between the resolver and the TLD authorities.

Finally, we find only 1 recursive resolver (124.219.23.209) in AS24154 in TW is poisoned because of collateral damage.

From the probing result, we can see that paths from recursive resolvers to root name servers seldom suffer from collateral damage, as the roots are heavily anycasted (except for the Chinese root servers), so DNS queries to the root seldom transit Chinese networks.

In contrast, the TLDs suffer from substantial collateral damage. We tested all of the 312 TLDs got from ICANN. For the three TLD in China (`.cn`, `.xn--fiqs8s`, `.xn--fiqz9s`), it is not a surprise that 43,322 (99.53%) resolvers return injected answers because the DNS resolution path have to get to the censored network.

Of greater concern is we find that 11,573 (26.40%) resolvers showed collateral damage for queries from one or more of 16 other TLDs. Figure 2 shows these TLDs and the number of affected resolvers. The second one, `.xn--3eb707e`, shares the same name infrastructure with the `.kr` ccTLD.

It seems strange that the number of affected resolvers for `.iq, .co, .travel, .no, .pl, .nz, .hk, .jp, .uk, .fi, .ca` are all around 90. We check the location of their name servers and find that it is not a coincidence: UltraDNS (AS12008) hosts the authority servers for all these TLDs.

Limited by space, we only present the detailed information for the most affected TLD: `.de`. As shown in Figure 3, over 70% of the resolvers from KR susceptible to collateral damage suffer collateral damage for `.de` queries, such as `www.epochtimes.de`.

Finally we constructed construct queries like KEYWORD-

AS Number	AS Name	Router IPs
4134	Chinanet	1952
4837	CNCGROUP China169 Backbone	489
4812	China Telecom (Group)	289
9394	CHINA RAILWAY Internet(CRNET)	78
9929	China Netcom Corp.	67
4808	CNCGROUP IP network China169 Beijing Province Network	55
9808	Guangdong Mobile Communication Co.Ltd.	38
17633	ASN for Shandong Provincial Net of CT	25
4538	China Education and Research Network Center	22
17816	China Unicom IP network China169 Guangdong province	19
Total 39 ASes		

Table 3: Information of top 10 injecting ASes.

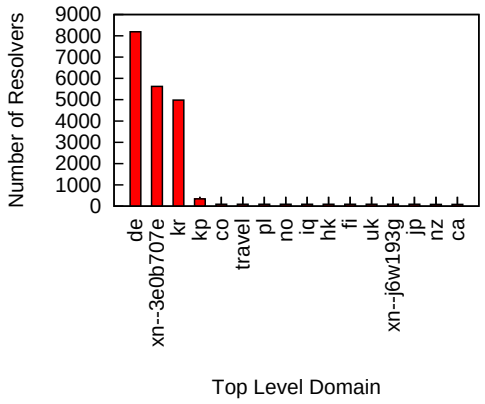
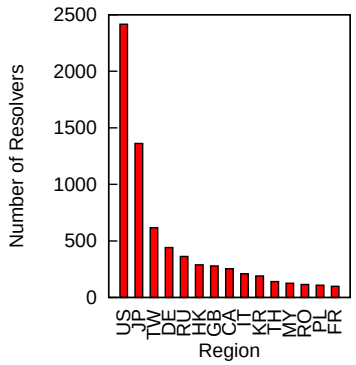
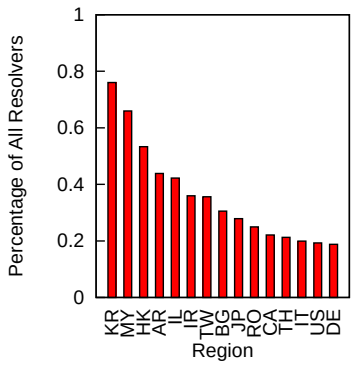


Figure 2: Affected TLD.



(a) Number of affected resolvers.



(b) Percentage of affected resolvers.

Figure 3: Distribution of affected resolvers for TLD `.de`.

.NXNAME.authority.tld (e.g., `www.twitter.com.abssdfds.ibm.com`) to explore paths from the resolvers to authoritative name servers for several domains.

We selected the top 82 popular domains from alexa.com, after excluding 18 Chinese sites. We see that queries for six domains could potentially trigger censorship on 30–90 resolvers, as shown in Figure 4. Although the number of affected domains and resolvers seem small comparing to the results of TLDs testing, this may represent the tip of the iceberg, considering the huge number of domain names of the whole Internet.

4.4 Further Analysis on Measurement Results

Table 5 gives the total number of resolvers suffering from collateral damage for root, TLDs and the top 82 domain names. 26.41% of experimental resolvers are polluted, distributed in 109 regions.

Unlike the worries presented by Mauricio [6], our measurement shows that the primary damage source arises from censored transit paths to TLD servers. According to Mauricio [6], the operator of I-Root server, Netnod, "withdrew their anycasted routes until their host (CNNIC) could secure assurances that the tampering would not recur". Our result partly confirmed their action. Since the roots themselves are highly anycasted, its unlikely that a path to a root needs to transit China.

In contrast, apparently a large amount of transit from the United States to Germany passes through China, resulting in the significant collateral damage to the `.de` ccTLD.

Rank	Region	Affected Resolver	Affected Rate
1	IR	157	88.20%
2	MY	163	85.34%
3	KR	198	79.20%
4	HK	403	74.63%
5	TW	1146	66.13%
6	IN	250	60.10%
10	IT	392	37.23%
14	JP	1437	29.39%
16	RU	835	25.26%
18	US	3032	24.22%
20	CA	272	23.65%
25	DE	470	20.04%
Total 109 Affected Regions			

Table 5: Collateral damage rate of different regions.

5. DISCUSSION

The cause of the collateral damage presented in this paper is the censorship activities by ISPs providing transit, not just connectivity. Although we'd hope otherwise, we believe it is naive to expect these ISPs to stop or avoid to applying DNS-injection based censorship activities, due to the significant social and political factors these ISPs face.

One possibility would be for the ISPs to apply more strict checks to avoid polluting transit queries. Although we do not support broad censorship activities, we hope that this

DNS Level	Affected Resolvers	Affected Rate
Root	1	0.002%
TLD	11573	26.40%
Authoritative	99	0.23%

Table 6: Number of affected resolvers in different level.

paper will raise awareness of the collateral damage caused by indiscriminate DNS censorship. If ISPs only act to censor customers, not transit, this prevents the collateral damage. However, because of the closed nature of the many censorship activities (such as the DNS filter in China), it is unclear to us if there are any technical challenges for those ISPs to implement such policy or not.

If the censoring ISPs do not change their current practice of DNS-injection, another possibility is for neighboring ISPs to consider them invalid for transit: the neighbors should prefer alternate paths and not advertise transit whenever an alternate path exists. In particular, the TLD operators should monitor their peering arrangements to check for censored paths.

Finally, and most importantly, DNSSEC naturally prevents this collateral damage, especially on the TLD level. Both the `.de` and `.kr` domains sign their results, enabling a DNSSEC-validating resolver which rejects the unsigned injected replies while waiting for the legitimate signed replies to avoid suffering collateral damage due to packet injection.

6. CONCLUSION

The contributions of this paper include:

(1) Comprehensive analysis of collateral damage by DNS injection. Iterative queries to different level of name servers, multiple name servers distributed in different locations and dynamic and anycast routing, are all factors which may cause a query to transit a censored network, even though both the user and the target are outside the censored area.

(2) Discovering and locating DNS injectors. We probed all the Internet to find the indiscriminate DNS injectors, locating these DNS injectors in 39 Chinese ASes.

(3) Measurement of affected recursive resolvers all over the world. We measured 43,842 open recursive resolvers in 173 countries, and found that 26.41% of them in 109 countries could be polluted.

(4) Primary path of pollution: from resolver to TLD servers. We find that the primary collateral damage arises from transit between the resolver and the TLD authorities, particularly the authorities for `.de` and `.kr`.

We expect to continue our study on the measurement of the collateral damage caused by DNS injection, using multiple vantage points and an expanded list of HoneyQueries. Although we has not come to a solution to allow recursive resolvers to be immune to the collateral damages from DNS-based censorship apart from DNSSEC validation, we hope our result can increase the Internet community's awareness of such behaviors, and take actions to actively detect and resist such pollution to the whole Internet.

7. REFERENCES

[1] M. A. Brown, D. Madory, A. Popescu, and E. Zmijewski. DNS Tampering and Root Servers, Nov. 2010. `http://www.renesys.com/tech/presentations/pdf/DNS-Tampering-and-Root-Servers.pdf`.

[2] R. Bush, M. Patton, R. Elz, and S. Bradner. Selection and Operation of Secondary DNS Servers. *RFC2182*, 1997.

[3] J. Crandall, D. Zinn, M. Byrd, E. Barr, and R. East. ConceptDoppler: A Weather Tracker for Internet Censorship. In *Proceedings of the 14th ACM Conference on Computer and Communications Security*, CCS'07, pages 352–365, New York, NY, USA, 2007. ACM.

[4] M. Dischinger, M. Marcon, S. Guha, K. P. Gummadi, R. Mahajan, and S. Saroiu. Glasnost: Enabling End Users to Detect Traffic Differentiation. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, NSDI'10, pages 27–27, Berkeley, CA, USA, 2010. USENIX Association.

[5] R. Elz and R. Bush. Clarifications to the DNS Specification. *RFC2181*, 1997.

[6] M. V. Ereche. Odd Behaviour on One Node in I root-server, 2010. `https://lists.dns-oarc.net/pipermail/dns-operations/2010-March/005260.html`.

[7] G. Lowe, P. Winters, and M. L. Marcus. The Great DNS Wall of China. `http://cs.nyu.edu/~pcw216/work/nds/final.pdf`, 2007.

[8] P. Mockapetris. Domain Names - Concepts and Facilities. *RFC1034*, 1987.

[9] P. Mockapetris. Domain Names - Implementation and Specification. *RFC1035*, 1987.

[10] C. Partridge, T. Mendez, and W. Milliken. Host Anycasting Service. *RFC1546*, 1993.

[11] L. Spitzner. Honeytokens: The Other Honeypot. `http://www.symantec.com/connect/articles/honeytokens-other-honeypot`, 2003.