Analyzing and Mitigating Privacy with the DNS Root Service

Wes Hardaker
USC/Information Sciences Institute
hardaker@isi.edu

Keywords

privacy, dns, root, analysis, leakage

ABSTRACT

Processing of all DNS requests start at the root of the DNS tree and make use of either cached data from previous requests, or by traversing the DNS tree for the missing information. When *QNAME minimization* is not in use, queries forwarded to the parental nodes in the DNS tree may leak private DNS query data. In this paper we examine 31 days during the month of January 2017 of queries sent from two recursive resolvers placed in two residential networks to the DNS root server operated by USC/ISI's, analyzing the leaked QNAMEs for an impact on the network's privacy. We then compare a few DNS privacy preserving techniques against the privacy analysis against these networks. Finally, we introduce a new solution called *"LocalRoot"* that enables users to entirely mitigate privacy concerns when interacting with the DNS root server system, while other solutions fail to completely protect users from all privacy analysis methods.

1. INTRODUCTION

Networking clients wishing to connect to Internet services typically start by sending Domain Name System (DNS) requests to recursive resolvers responsible for handling the requests and returning answers back to the clients. The resolution process is a multi-step problem that involves querying the portions of the DNS responsible for each portion of a domain name (like *www.example.com*). For performance, bandwidth and other reasons, DNS records are generally cached for a period of time so that repeat lookups don't result in more network traffic. Thus, once the DNS servers for *example.com* are known, recursive resolvers need not traverse the tree from the beginning and can instead immediately send a new query for *images.example.com* to the correct *example.com* name server, until relevant cache timers expire. The start of the DNS resolution system begins with the "Root Server System", which is responsible for answering queries about where (and whether) Top Level Domain servers exist for a given client query. The architecture of this query process is depicted in Figure 1. The leakage of DNS requests to the Root Server System is the focus point of this paper.

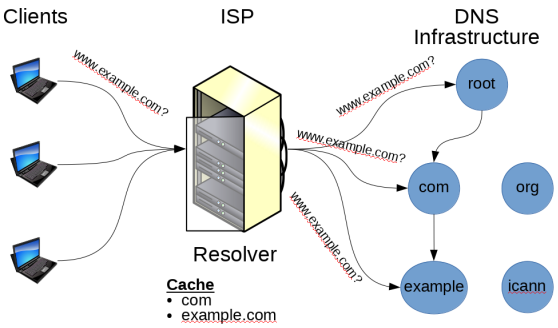


Figure 1: Domain Name System Resolution Process

Most DNS recursive resolvers today send the entire requested DNS name (e.g. *www.example.com*) unencrypted to all the servers it needs to correspond with, as it is unknown to them where in the name the different resolution zone cuts exist. Only recently has the concept of DNS Query Name Minimization [1] been explored to prevent these names from leaking to higher levels of the DNS tree than is necessary. Additional, methods to encrypt the transport have been undertaken [9, 12] in an effort to protect against man-in-the-middle attacks between clients and recursive resolvers.

What has been unclear to date is the level of privacy leakage occurring because of only occasional (cache miss) requests to the parental zones in the DNS tree, especially in light of the implicit caching and query spreading of the requests. To study this, we undertake an rough analysis of a months worth of DNS records sent from two residential networks (§2) to the root server system run by USC/ISI ("*B-Root*") in §3. This allows us to study the how privacy is affected by both caching (which prevents all new requests from departing until the cache expires) and query spread (by not examining the data sent to other root server operators). We then compare various techniques for protecting DNS privacy in §4 and

	Dataset	
	RES1	*RES2*
IPv4 Packet Count	52191	2049
IPv6 Packet Count	27675	9
Total	79866	2049

Table 1: Dataset details

simultaneously introduce a new root zone caching infrastructure called "LocalRoot" (§4.3) that alone provides complete DNS privacy protection for use with the root zone.

2. DATASETS FOR PRIVACY ANALYSIS

We select datasets from two residential households that have deployed local recursive resolvers within the entire month of January, 2017. Further details about these networks are reserved for the *Ground Truth* sections of the paper later on.

Because the author of this paper is at least partially familiar with the usage of these networks, this study can't be considered a truly unbiased and blind analysis. However, we choose analysis techniques that extract data in mostly generic ways to avoid preemptive data selection as much as possible, though our potential conclusions from the analysis may still be biased with knowledge. Our primary goal, however, is not to demonstrate successful privacy extraction techniques but rather to demonstrate that potentially privacy leaking data actually can be found within DNS queries, and some level of prior knowledge helps us toward this goal.

The high-level statistics of the resulting dataset are shown in Table 1. Note that these packet counts include both queries and responses.

2.1 Ethical Considerations

The owners of the two residential networks (of which the author is one) being studied (*RES1* and *RES2*) explicitly gave consent for their network DNS data to be used in this study. We also selected USC/ISI's root server instance, since it is also under our control. Thus, both sides of the communication path have agreed to participant in this study. We are not releasing the content of these datasets publicly, however, since a complete line-by-line analysis of the data has not been done to ensure no inappropriate third-parties would be affected.

3. PRIVACY ANALYSIS

An entire book could be written about the ways in which DNS data could be broken down for privacy leakage. We select a few broad stroke analysis techniques to study the datasets in bulk, demonstrating that even high-level analysis can reveal network structures and usage (§3.1, §3.2), geographical information (§3.3), and temporal usage patterns (§3.4). We additional touch upon the process of doing deeper dives into data with special searches of the leaked QNAMEs (§3.5) that reveals even greater detail of both hardware and software in use within the networks.

In each analysis we separately analyze the data from both *RES1* and *RES2* and include ground truth descriptions for comparison.

3.1 Internet Protocol Version Analysis

The first obvious question comes from looking at the breakdown of the dataset shown in Table 1: *What conclusions can we draw from the different usage of IPv4 vs IPv6?*

***RES1* Analysis:** *RES1* shows a significant difference between traffic levels sent over IPv4 vs IPv6. There could be a number of reasons behind this, but the most likely may be the simplest: the IPv4 connectivity to USC/ISI's root server performs better than the IPv6 connectivity.

***RES1* Ground Truth:** The *RES1* network does have both IPv4 and IPv6 connectivity, with the IPv6 connectivity being provided via a IPv6-over-IPv4 tunnel to Hurricane Electric. And traffic through a tunnel, with its introduced extra hops, would be expected to provide a slower service.

***RES2* Analysis:** *RES2* shows only IPv4 connectivity, at least with respect to packets sent to the root server.

***RES2* Ground Truth:** Indeed, only IPv4 is deployed within this network.

General Conclusions: This simple starting analysis may seem trivial to even note, but even IP connectivity can be revealing about what networks a resolver is connecting over when combined with other analysis.

3.2 RRTYPE Analysis

We next study the breakdown of different Resource Record Types (RRTYPEs) of the queries sent from the two network resolvers to the root server for all traffic, shown in Figure 2. The results show the clear (expected) bias toward address records being the most common type of lookup for both networks.

***RES1* Analysis:** The other values of RRTYPEs in the *RES1* dataset provide much more interesting data to analyze and pose some interesting questions: *Why are SOA and TXT records primarily queried over only IPv4? Why are the number of NS records double the number of SOA records?*

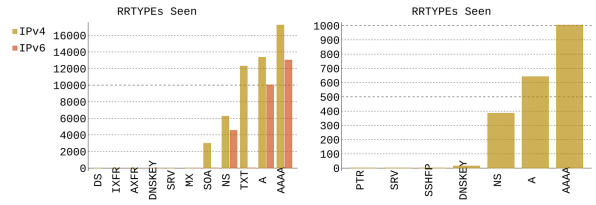
If we look at the dataset for just the query rates of SOA and TXT records (Figure 3), we quickly discover a fairly flat rate of 2 and 16 queries and associated responses per hour for SOA and TXT types respectively, with an unusual higher burst of SOAs centered near January 5th. The only reasonable reason for a near-constant repetitious query pattern is likely the existence

of a DNS monitoring system within the network. Looking at the TXT queries, nearly all of them (99.43%) are for either *hostname.bind* and *id.server* (common ways to query for a server's instance identifier) and *version.bind* and *version.server* (common mechanisms for querying for a name server software's version number). Considering this as a clue, we consult the database for public RIPE Atlas probes and find the IP address listed there. This can suddenly expose an additional significant quantity of information about the IP address, including it's physical position based on the registered GeoIP data (locating it in Davis California in the United States), network connectivity information (e.g. ASN number), connection up/down times, etc, which is a pretty significant cache of information. Because the IP address of study is directly in the RIPE database, we can also conclude that the RIPE Atlas probe is behind a IPv4 NAT.

This leaves only 70 remaining TXT records out of an original 12321 . Of those remaining, 12 are requests looking for _domainkey records, which are used by mail servers (and begins our suspicion that a mail server is operating within this network).

We thus consider eliminating all queries for the root itself to further study RRTYPEs without the monitoring system influence. The results, shown in Figure 4, show an even larger bias toward mostly address and text record lookups. We eliminate those dominating record types in Figure 5, and which provides us a few hints about other uses of the network. The existence of MX RRTYPEs clearly indicates the presence of a mail server and the AXFR and IXFR types likely indicates that the presence of a authoritative DNS slave server. The presence of a SRV RRTYPE indicates a potential number of application types that could be deployed on the network, looking quickly at them show a breakdown between two primary prefixes: nine *_minecraft_tcp* and one *_xmpp-server._tcp*, which quickly gives us insight into the use of at least the Minecraft game and a XMPP (jabber) protocol client.

***RES1* Ground Truth:** This network does indeed host a fairly well used mail server, responsible for hosting a number of fairly moderate sized mailing lists (order 1000s of subscribers). There is a DNS monitoring system deployed in the household: a RIPE Atlas node is connected to the network and regularly sends queries to all the root servers, including B, among with the other various measurements performed by the Atlas networks. The residence is indeed in Davis, although the address in RIPE is not perfectly centered on the actual residence but rather a near-by house. Minecraft is indeed played heavily in the household, and there is a fair amount of jabber; it is interesting to note the xmpp protocol applications are used more than the minecraft game but the quantity numbers seen show the reverse. This is likely from the fact that the jabber clients are much longer lived and shouldn't need frequent lookups, where as the minecraft client is constantly making connections to new servers.



(a) *RES1* (b) *RES2*

Figure 2: RRTypes Seen

***RES2* Analysis:** The *RES2* dataset shows a much more clear trend toward being a mostly end-user based DNS lookup resolver, as the trend is even more heavily biased toward address space. Looking at the Figure 5 figure we can see that there is some use of both SRV and SSHFP records, showing the potential of another application-type mining and the use of SSH. When we look into the data for the SRV record, we find that the QNAME was actually for *10.0.0.38* which is an odd (malformed?) QNAME for an SRV query, but it does disclose the existence of an internal private RFC1918 address space in use of 10.0.0.0/8, along with one likely internal address (*10.0.0.38*). If we look into the SSHFP request, we find it for an odd name of just *"g2"*, which could be a typo or potentially the name of an internal host. This further shows us that names of internal hostnames can frequently leak to the root zone because SSH clients issue queries for SSHFP records with and without default search domains.

***RES2* Ground Truth:** Interestingly, although the private address space of 10/8 is used within the network, there is not normally anything deployed within the 10.0/16 section of that address space. Thus, it is unclear where the odd request for the *10.0.0.38* SRV record came from. There is also an internally named host called *"g2"*, proving that the guess of an internal network name discovery was a success.

General Conclusions: Analyzing RRTYPEs proves to provide useful leading directions that can be quickly followed by more in depth studies based on the leads. In particular, we showed that with minimal effort these initial directions quickly lead to significant disclosures of privacy in various records, especially with respect to applications running within a network.

3.3 Geographical Analysis

Geographical connectivity analysis can be performed on DNS resolution data by examining the *Country Code*

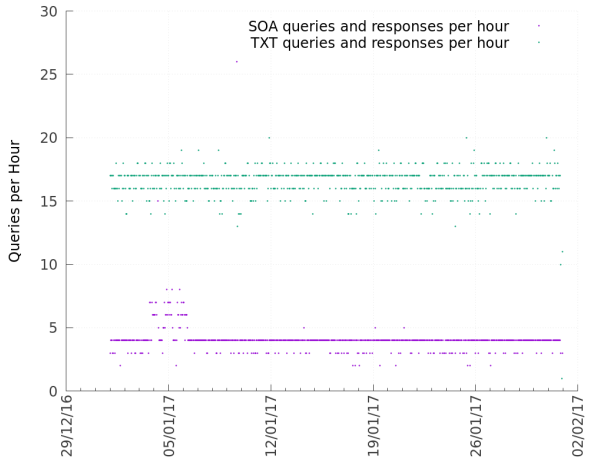
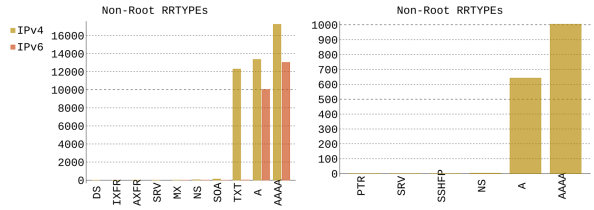


Figure 3: SOA Query Rates Per Hour for *RES1*



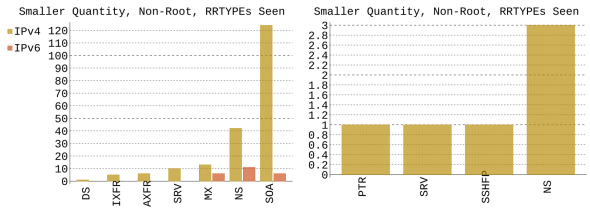
(a) *RES1* (b) *RES2*

Figure 4: RRTYPES seen for queries to zones other than the root

TLDs (ccTLDs) represented in all the requests. Extracting them and mapping them to a heat map shows a view of the most queried ccTLDs mapped back to the country of origin in the maps in Figure 6 and the top 10 country codes seen in Figure 7.

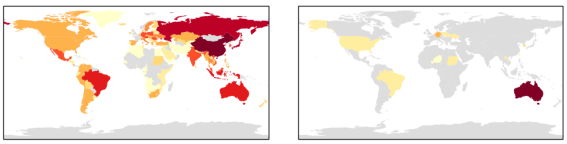
***RES1* Analysis:** Simple visual analysis shows the top three ccTLDs in Figure 7 are from China, Japan, and Russia, giving the impression that the resolver may be in the Asian region or that its primary correspondence may be with entities in those regions.

***RES1* Ground Truth:** The reality is that very little "real" communication happens with the countries in those regions. The mail server, however, does receive a very large quantity of spam [7] from countries in that region, most of which is filtered automatically by spamassassin [5] and other tools. The result is that the *RES1* network owner was rather surprised by this map, as it didn't match the expected usage at all. An important conclusion we can draw from this analysis vs ground truth is that mail servers can obscure the truth because of the world's spam level (The 2009 spam analysis of this network in [7] showed a 70% auto-drop rate of all email messages arriving at the server).



(a) *RES1* (b) *RES2*

Figure 5: Magnified smaller RRTYPES seen for queries to zones other than the root



(a) *RES1* (b) *RES2*

Figure 6: Geographical map of highest percentage ccTLDs

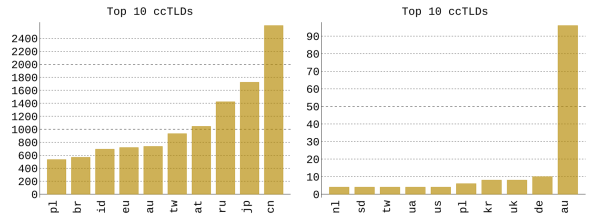
***RES2* Analysis:** The map and bar graphs for *RES2* shows communication for that network happening overwhelmingly with Australia, leading to a similar potential conclusion that may indicate where the server is housed or with whom the residents communicate.

***RES2* Ground Truth:** The owner of this network is also somewhat mystified with the results, as there isn't a strong tie to Australia at all within the household.

General Conclusions: As noted, spam does a good job of masking actual network usage. But another important consideration when analyzing ccTLD usage is that the usage of ccTLDs within the various countries is not consistent and will skew any analysis. For example, within the United States the usage of the *.us* ccTLD is not very common, but in many other countries the perceived usage of their ccTLDs is much higher (e.g. *.cn*). The authors don't know of a usage study of ccTLD popularities within countries, however, which might help applying a weight to the data to drive more accurate results. Additionally, as different countries make use of different Time To Live (TTL) values – the maximum cache timing length – it would be appropriate to also weight the request frequency based on the inverse normalized value of TTLs.

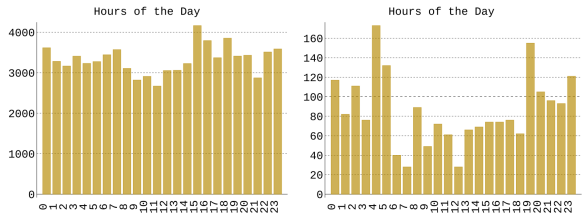
3.4 Temporal Analysis

Both general and temporal traffic analysis has been well shown to be a problem that encryption alone can not solve [13]. We examine the temporal nature of the data in our datasets by examining the hours when the



(a) *RES1* (b) *RES2*

Figure 7: Top 10 ccTLDs Seen



(a) *RES1* (b) *RES2*

Figure 8: DNS Traffic per Hours

DNS messages are transmitted (in GMT) in Figure 8. We also examine the traffic per day of the week in Figure 9 (*0 signifies Sunday*) for the last 28 days of the month (we chose the last 28 days to avoid January 1st, a common holiday).

***RES1* Analysis:** The 7 hour range from 8 to 14 in Figure 8 appears slightly lower, potentially indicative of a "sleep period". Assuming a typical night based sleep period, then we might be able to infer a timezone offset of roughly 6-8 hours from GMT. Even if correct, it is noteworthy that a large number DNS requests still occur frequently during all periods of the day with less variation than one might expect from a household that might be empty in the daytime – possibly indicating a household that typically not vacant at all.

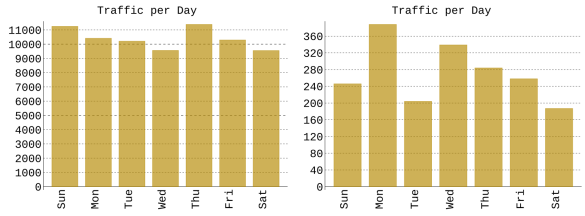
For the days of the week graph (Figure 9) we note that there isn't a huge variation in traffic per day, with thus little conclusion that can be drawn from the graph other than it reinforcing the conclusion of always populated we drew from the hourly graph. Or there are enough continuously active internet devices (such as the expected mail server) that we can not expect to see much variation per-day in the first place.

For a household where the occupants worked Monday to Friday (for example) with little else going on in the network, we may be able to extrapolate both working days and working hours from enough data.

***RES1* Ground Truth:** The residence is indeed in the Pacific Time Zone, as we already revealed above, which is indeed 7 hours off of GMT in January. At least some residents are typically home due to a work-at-home type arrangement.

***RES2* Analysis:** The *RES2* Figure 8 graph also shows a reasonably sized drop over a longer range of time between the GMT hours of 6-18, which establishes a potentially similar but longer sleep-wake cycle. Because it seems extended but starts near the same time, it is worth noting that the median appears shifted upward in time by an hour or two bring it closer to a potential 4-6 hour offset from GMT.

The *RES2* graph in Figure 9 seem even more significantly random, with no discernible weekday/weekend



(a) *RES1* (b) *RES2*

Figure 9: DNS Traffic per Week Days

type pattern appearing. The only conclusion is that the network likely receives regular use that does not vary based on a daily pattern.

***RES2* Ground Truth:** The owner of the residence telecommutes from home, and thus the daily graph's lack of variation is to be expected. The residence is on the east coast of the United States, with a timezone shift of 4 hours off of GMT in January, which is reasonable close to our guess.

General Conclusions: It appears that temporal analysis of DNS data even to the root-servers may be enough to provide rough estimates of sleep-wake cycles within residential networks. Note that though we guessed at a typical sleep cycle of roughly 10pm to 6am local time, many variations from that are common and those time ranges often shift by a few hours depending on life style. Less common is graveyard shifts, which could easily throw off hour-based temporal analysis but are also statistically less likely. We might find more accurate days-of-the-week results if we examined a sample size that included more than four weeks. It is worth noting that the month of January has a number of holidays that occurs on Mondays within the month as well.

3.5 QNAME Analysis

We analyze the QNAMEs in the dataset by breaking them down in a number of ways: 1) the Top Level Domains (TLDs) of the QNAMEs, 2) the Second Level Domains (SLDs) of QNAMEs, and 3) separating QNAMEs based on the Public Suffix List (PSL).

TLD	Raw Count	Percentage
uk	327	0.41
my	398	0.50
de	437	0.55
vn	440	0.55
pl	533	0.67
br	568	0.71
id	693	0.87
eu	718	0.90
au	735	0.92
tw	932	1.17
at	1044	1.31
org	1224	1.53
edu	1396	1.75
ru	1423	1.78
server	1483	1.86
jp	1723	2.16
cn	2596	3.25
bind	10768	13.48
com	14843	18.58
net	17448	21.85

Table 2: Top 20 TLDs seen at *RES1*

TLD	Raw Count	Percentage
tfioqfwcsluo.	2	0.10
vlussrjhj.	2	0.10
br	4	0.20
la	4	0.20
mail.	4	0.20
nl	4	0.20
sd	4	0.20
tw	4	0.20
ua	4	0.20
us	4	0.20
z	4	0.20
pl	6	0.29
edu	7	0.34
kr	8	0.39
uk	8	0.39
de	10	0.49
au	96	4.69
net	316	15.42
org	380	18.55
com	743	36.26

Table 3: Top 20 TLDs seen at *RES2*

3.5.1 Top Level Domain Analysis

Although we already performed some analysis of TLDs within §3.3, we turn to a more complete analysis and look at all TLDs, not just the ccTLDs. The top 20 TLDs of *RES1* and *RES2* are listed in Table 2 and Table 3, respectively.

***RES1* Analysis:** In this data we find an unexpected fact: the most popular TLD is ".net", when we might have assumed the all prevalent ".com" would have been the most popular. We will wait to dive into SLD analysis to determine possible causes. We also note that the "bind" and "server" names, which are not an actual TLDs, comes up in third and fourth places, respectively. This is certainly due to the RIPE atlas probe within the network that is querying for names like *hostname.bind* and *id.server*, per discussion in §3.2. We also notice that the ccTLDs ".cn", ".jp" and ".ru" are high in the list, which is also to be expected based on the analysis in §3.3, though we do take note that those three TLDs are actually both higher in usage than ".org" and ".edu".

***RES2* Analysis:** The top 20 domains for the *RES2* dataset produce a more expected spread of popularity, with *.com*, *.net* and *.org* appearing in the top three spots in the expected order. Following that we see the previously discussed *.au* and *.de* zones. We do observe that the *.com*, *.net* and *.org* and org trio are significantly more popular, however. Finally, we note the oddity of the *.z* non-existence suffix, the internal-like *.mail* suffix (potentially hinting at an internal mail server) and two random string names that appear to be broken queries.

General Conclusions: There aren't many conclusions that we can draw from TLD analysis alone, other than the odd discrepancies noted above. It's possible this analysis would be better paired with deeper dives, which we attend to next.

3.5.2 Second Level Domain Analysis

We now exmaine the next level down to a hopefully much more enlightening lower-level details, and study which Second Level Domains (SLDs) are the most popular in the two residences. The top 20 SLDs are listed in Table 4 and Table 5.

In both of these datasets we note a few interesting names that appear visually interesting: a large number of SLDs seem to be other public suffixes [6] (e.g. *net.cn*, *ne.jp*, *co.jp*, *net.au*, etc) and thus deeper analysis is needed (see §3.5.3).

***RES1* Analysis:** Repeated queries to *msft.net* may indicate the presence of microsoft based software. Looking into the data, we find all of the queries to **.msft.net* are for *ns[1-4].msft.net* showing that some domain is using microsoft's DNS servers. There were far fewer queries for microsoft.com (12), indicating that some other domain with DNS hosting at *msft.net* is more likely the cause.

High levels of *returnpath.net* (an email delivery company) and *sorbs.net* (a "Spam and Open-Relay Blocking System") likely continue to further solidify our notion a mail server operating within the network. In particular, we wonder if *sorbs.net* might provide interesting data because typical lookups within it are in the form of *IPADDRESS.dnsbl.sorbs.net*, which would disclose the IP addresses of the machines that were attempting mail delivery to the target *RES1* network. However, when looking through the dataset we found that only requests for name servers (e.g. *A* and *AAAA* lookups for *nsNUM.sorbs.net* and thus find that no further privacy leakage actually occurred.

***RES1* Ground Truth:** There are actually no microsoft based OSes operating with any regularity within the network and it is unclear to date why so many requests are going to msft.net, showing that the conclusion of other domains using msft.net for DNS hosting is likely.

***RES2* Analysis:** A number of interesting domains appear in this list. The largest number of queries was for *vipcam.org*, showing the likely usage of a web-base camera within the residence. A quick search of "vipcam" shows both an Android app [3] and a iOS app [4].

We also note that both of the top two, *vipcam.org* and *shifen.com* are registred to companies in China. We also

sld	raw count	percentage
gatech.edu	444	0.84
sorbs.net	447	0.84
verisigndns.com	451	0.85
register.com	454	0.85
nhs.net	473	0.89
usg.edu	492	0.93
edu.tw	502	0.94
net.id	519	0.98
gtei.net	537	1.01
ntt.eu	553	1.04
tislabs.com	573	1.08
apnic.net	575	1.08
net.au	606	1.14
co.jp	608	1.14
returnpath.net	692	1.30
com.cn	693	1.30
ne.jp	758	1.43
anexia.at	939	1.77
net.cn	1266	2.38
msft.net	1475	2.78

Table 4: Top 20 SLDs seen at *RES1*

SLD	Raw Count	Percentage
fedoraproject.org	16	0.99
samsung.com	16	0.99
surriel.com	17	1.05
linode.com	21	1.30
iqnection.com	22	1.36
rpmfusion.net	22	1.36
msedge.net	25	1.54
root-servers.net	26	1.61
azuredns-cloud.net	32	1.98
sosdg.org	35	2.16
novell.com	42	2.59
google.com	44	2.72
dreamhost.com	54	3.34
verisigndns.com	54	3.34
amazonaws.com	75	4.63
msft.net	91	5.62
net.au	96	5.93
tislabs.com	103	6.36
shifen.com	204	12.60
vipcam.org	260	16.06

Table 5: Top 20 SLDs seen at *RES2*

note the high use of DNS hosted services at *msft.net*, *amazonaws.com*, and *verisigndns.com* as well as the DNS, web and e-mail hosting provider *dreamhost.com*.

The listing of *rpmfusion.net* likely indicates the presence of a rpm-based linux system, and the lowest entry of *fedoraproject.org* further points us toward the use of at least one Fedora linux based system. The presence of *msedge.net* in the list indicates the almost certain use of Microsoft Windows, likely with one of Windows 10 telemetry services enabled [10].

***RES2* Ground Truth:** The *RES2* residence does have IP cameras deployed in the household, but the owner is unsure why *vipcam.org* would be a destination DNS name, as the vipcam apps are not used. The network also does make use of the Fedora Linux operating system on a number of machines.

General Conclusions: We note that both authors worked recently for the company that uses the *tislabs.com* domain for email, and thus the appearance of that in the list of domains is not surprising. This is one of the few analysis points that indicates a possible relational tie between *RES1* and *RES2*, hinting that a greater differential analysis between the two datasets might produce interesting conclusions in a future study.

3.5.3 *Public Suffix List Analysis*

We turn to making use of the *Public Suffix List (PSL)* [6] to help split the data at public DNS registration points, the most likely boundary between organizations registering for a domain and the upper level DNS provisioning infrastructure. The results in Table 6 and Table 7 provide us a different view than the SLD list (§3.5.2).

name	raw count	percentage
a5.com	388	0.49
linode.com	406	0.51
cloudflare.com	424	0.53
telkom.net.id	436	0.55
gatech.edu	444	0.56
sorbs.net	447	0.56
verisigndns.com	451	0.56
register.com	454	0.57
nhs.net	473	0.59
usg.edu	492	0.62
gtei.net	537	0.67
aridns.net.au	550	0.69
ntt.eu	553	0.69
nintendo.co.jp	559	0.70
tislabs.com	573	0.72
apnic.net	575	0.72
vips.ne.jp	612	0.77
returnpath.net	692	0.87
anexia.at	939	1.18
msft.net	1475	1.85

Table 6: Top 20 domains under public suffix list zone code points seen at *RES1*

This analysis drops a number of items from the lists in §3.5.2, which were clearly registration points. This shows the importance of understanding where registration points in the DNS tree exist in order to accurately identify true destinations being used during an analysis, leaving us able to concentrate on the communication to real content or service providing zone names.

***RES2* Analysis:** Amongst the previously analyzed names of interest (e.g. *msedge.net*) we see *samsung.com* appear in the *RES2* dataset.

***RES2* Ground Truth:** Multiple samsung based devices are in use within this residence, confirming that we can detect types of hardware in use via DNS query names.

General Conclusions: We note a wide range of data center provider domains (e.g. *amazonaws.com*, *sosdg.com*, and *linode.com*) and common DNS providers (e.g. *cloudflare.com*, *msft.net* and *azuredns-cloud.net*) that gives a hint toward the world's popularity of centralized Internet services.

A point of future study might be to collect a series of common destination manufacturer domains to use as a index for collecting device usage based on DNS query names.

3.5.4 *Special Name Analysis*

There are a few "special" type name patterns that can be found within the DNS, such as with the special "_" character prefix (typically indicating a special protocol lookup).

***RES1* Analysis:** Searching for "_" turned up an interesting set of hosts, the results of which for *RES1* is shown in Table 8.

The successful search for names containing "_" showed

Name	Raw Count	Percentage
samsung.com	16	0.78
surriel.com	17	0.83
linode.com	21	1.02
iqnection.com	22	1.07
rpmfusion.net	22	1.07
u2.amazonaws.com	22	1.07
u1.amazonaws.com	24	1.17
msedge.net	25	1.22
root-servers.net	26	1.27
azuredns-cloud.net	32	1.56
sosdg.org	35	1.71
novell.com	42	2.05
google.com	44	2.15
dreamhost.com	54	2.64
verisigndns.com	54	2.64
msft.net	91	4.44
aridns.net.au	96	4.69
tislabs.com	103	5.03
shifen.com	204	9.96
vipcam.org	260	12.69

Table 7: Top 20 domains under public suffix list zone code points seen at *RES2*

another point in favor of a mail-server running within the network because of the names containing "_domainkey", a security mechanism for E-Mail delivery and reputation. These names are potentially indicative of what systems are delivering mail to the machine, exposing a portion of the social network associated with this residence.

The "_minecraft._tcp" reveals two interesting elements: 1) the fundamental presence of someone who plays minecraft enough to trigger queries to the root zone and 2) the disclosure of three IP addresses, two of which are likely local internal addresses from RFC1918 [14] private address space (*10.0.0.2* and *10.0.0.18*). The third IP address is a global a address showing that someone within the *RES1* residence plays minecraft with someone on a comcast network with a dynamic IP address (*73.41.83.66*). Finally, the reference to an "_xmpp" based name indicates the presence of a XMPP (jabber) client on the network. The complete list in Table 8 is well worth reading for the humor of some of the other names found within it.

***RES1* Ground Truth:** 10.0.02 and 10.0.0.18 are indeed addresses statically assigned to the two machines internally to *RES1*. As noted in §3.2, Minecraft and jabber are applications both frequently in use within the *RES1* network as well.

***RES2* Analysis:** Searching for the "_" character in *RES2* turned up zero hits, so no analysis was available to be done.

4. MITIGATION OPTIONS OF ROOT LEAKAGE

We next examine three existing privacy preserving techniques, *QName Minimization* (§4.1), *TLS based DNS encryption* (§4.2) and *LocalRoot* (§4.3), for how they may mitigate the concerns brought up in §3.

4.1 DNS Query Name Minimization

QName Minimization is defined in RFC7816 [1] ("DNS Query Name Minimisation to Improve Privacy"), and encourages recursive resolvers to send only Name Server (NS) record requests to the parental level of the DNS tree. Thus, instead of sending *www.example.com* to the root servers, it would instead only request *com*'s NS records from the root, and then only *example.com*'s NS records from *com*. This stops the leakage of the entire domain name being leaked to both *com* and the DNS root servers. This effectively stops many, but not all, of the privacy analysis attacks described in §3. Specifically, because query minimization does not stop requests from traversing the network, temporal and traffic analysis may still be effective. Additionally, any study of traffic involving TLDs (including ccTLDs) would still be unimpeded by query minimization.

Name	Raw Count
_adsp._domainkey.ihjqljmo.cc.	1
_adsp._domainkey.linkedin.chi.namibia.na.	1
_adsp._domainkey.newsbank.club.	2
_adsp._domainkey.till.name.	1
_adsp._domainkey.user1-computer.i-did-not-set *(cont...)*	
-mail-host-address-so-tickle-me.	1
_adsp._domainkey.uzps.co.sy.	1
_adsp._domainkey.xtreamues.trade.	2
_minecraft._tcp.10.0.0.18.	4
_minecraft._tcp.10.0.0.2.	1
_minecraft._tcp.73.41.83.66.	4
_xmpp-server._tcp.pandion.im.	1
dkim._domainkey.speedbring.win.	1
libglesv1_cm.so.	2
mesmtp._domainkey.mad.paris.	1
postfix._domainkey.luffy.cx.	1
testglxgetprocaddress_genentry.sh.	2
testpatchentrypoints_gldispatch.sh.	2

Table 8: Names in *RES1* containing the "_" character

4.2 TLS based DNS encryption

The (D)TLS based solutions from RFC7858 [9] and RFC8094 [12] are currently designed only to protect client to resolver communications, but for purposes of completeness we assume that eventually channel-based encryption may be applied to all DNS transactions, such as the resolver to authoritative server communication being studied in this paper. We consider the case of complete deployment, which will likely take decades to complete, and note that it provides a partial solution to the analysis techniques from §3. Specifically, though it mitigates man in the middle attacks well, it does not mitigate against potentially compromised or malicious root server instances and is only given a rating of *Partial (P)* in those cases. It also fails to protect against IP version and temporal analysis, like QNAME minimization.

4.3 LocalRoot: Serving the Root Zone Locally

RFC7706 [11] (" Decreasing Access Time to Root Servers by Running One on Loopback") documents how to keep a cached a copy of the root zone within a recursive resolver, to *"provide faster negative responses to stub resolver queries that contain junk queries, and to prevent queries and responses from being visible on*

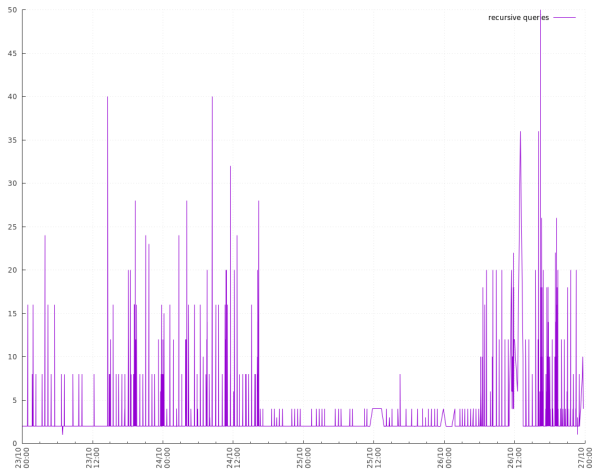


Figure 10: A graph of DNS requests to USC/ISI's root server showing that as *LocalRoot* is enabled, the number of requests to B-Root significantly drops

the network". At USC/ISI, we have created a (free) RFC7706 configuration generation system and paired it with a DNS notification service to allow recursive resolvers to become proper slaves of the DNS root server system. This service, called *LocalRoot*, exists at `localroot.isi.edu` and protects against all of the analysis privacy attacks from §3 by removing the need to contact the root servers entirely during normal query processing. (In previous work [8] we have already shown the benefits of caching systems and their ability to preserve DNS privacy.) Pre-caching the root zone data comes at the expense of needing to routinely transfer the entire root zone to the resolver; although the root zone is a relative small zone it is likely the total bandwidth will be higher than requesting only the needed records. The effect of enabling and then later disabling a LocalRoot cache in *RES1* can be seen in Figure 10. Note that in the period of time when *LocalRoot* is enabled, some requests still go out for two reasons: 1) the root server cache will occasionally send out SOA queries to see if it's slave copy is up to date and 2) the RIPE Atlas probe discovered in this network (§3.2) is still sending measurement probes directly to the root server.

4.4 Solution Comparisons

If we compare the privacy preserving success of *QName Minimization* (§4.1), *TLS based DNS encyrption* (§4.2) and *LocalRoot* (§4.3), we find that though QName Minimization provides protection against all levels of the DNS, and TLS based communication provides adequate protection against man-in-the-middle attacks, only the complete caching provided by *LocalRoot* protects against every form of analysis from §3.

Analysis Method	Sec.	QName Min. §4.1	TLS §4.2	LocalRoot §4.3
IP Version Analysis	§3.1	N	N	Y
Record Type Analysis	§3.2	Y	P	Y
Country Code Analysis	§3.3	N	P	Y
Temporal Analysis	§3.4	N	N	Y
Top Level Domain Analysis	§3.5.1	N	P	Y
Second Level Domain Analysis	§3.5.2	Y	P	Y
Public Suffix List Analysis	§3.5.3	Y	P	Y

Table 9: Analysis method v.s. solution protection effectiveness

The success of both techniques against each analysis method is shown in Table 9.

5. CONCLUSIONS

We've shown that a number of high level analysis techniques can be used to quickly extract interesting information from data sent to parental authoritative servers in the DNS system. These techniques can expose data about an underlying network and the people that use it. When determining what types of attacks to launch against a network, knowing the types of devices (e.g. IP cameras), operating systems (e.g. *Fedora Linux*) and applications in use (e.g. Minecraft and VipCam) can be highly useful information. Since collecting and examining DNS traffic can be done via passive collection methods, this network analysis can be done in stealth without detection by the network operator that might otherwise notice techniques like port scanning.

There are a number of DNS privacy preserving techniques, each of which can combat various aspects of DNS privacy attacks. Finally, we introduced *LocalRoot* as a root name server caching solution.

6. FUTURE WORK

This starting body of work could easily extend in multiple directions. First, we only scratched the surface of privacy analysis techniques that can be applied to a large collection of DNS requests to and from parental level authoritative servers; thus a significant amount of work remains in extending and adding to these analysis techniques. Second, resolvers from other network types could be studied, such as enterprise or ISP networks. Third, the mechanisms used here for analysis could be encoded for rapid analysis of datasets for use in bulk comparisons of DNS data. This would lead to the ability to do differential analysis of data sets from multiple networks.

We believe that technologies like *LocalRoot* can be used for more zones than just the root zone. Based on received feedback, effort may start to study how it might be used for various TLDs, for example and we may look into using data sources like the TLDR [2] effort provides.

7. ACKNOWLEDGMENTS

The author would like to thank Robert Story (an USC/ISI employee) for his willingness to contribute a dataset. Wes Hardaker's work in this paper is partially supported by USC as part of B-Root research activity.

8. REFERENCES

[1] S. Bortzmeyer. DNS Query Name Minimisation to Improve Privacy. RFC 7816 (Experimental), March 2016.

[2] Mathew Bryant. Tld records. http://github.com/mandatoryprogrammer/TLDR.

[3] IP cam. vipcam android app. https://play.google.com/store/apps/details?id=com.psd.vipcam&hl=en.

[4] IP cam. vipcam ios app. https://itunes.apple.com/us/app/vipcam/id1092024204?mt=8.

[5] Apache Foundatation. Apache spamassassin. https://spamassassin.apache.org/.

[6] Mozilla Foundation. Public suffix list. https://publicsuffix.org/.

[7] Wes Hardaker. I've got mail! http://pontifications.hardakers.net/computers/ive-got-mail/.

[8] Wes Hardaker. Changing dns usage profiles for increased privacy protection, February 2017.

[9] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman. Specification for DNS over Transport Layer Security (TLS). RFC 7858 (Proposed Standard), May 2016.

[10] james4264. Massive uploads to msedge.net. https://community.spiceworks.com/topic/1954402-massive-uploads-to-msedge-net.

[11] W. Kumari and P. Hoffman. Decreasing Access Time to Root Servers by Running One on Loopback. RFC 7706 (Informational), November 2015.

[12] T. Reddy, D. Wing, and P. Patil. Dns over datagram transport layer security (dtls). RFC 8094, RFC Editor, February 2017.

[13] Haya Shulman. Pretty bad privacy: Pitfalls of dns encryption. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society*, WPES '14, pages 191–200, New York, NY, USA, 2014. ACM.

[14] C. Weider, J. Fullton, and S. Spero. Architecture of the Whois++ Index Service. RFC 1913 (Historic), February 1996.