Using DITL data to look at leaked queries

Paul Hoffman

DNS-OARC 28
9 March 2018

Overview

- Processing the DITL 2017 data
- Winnowing down the data on TLDs seen
- DNS-OARC systems for analysis
- Analysis on a single root operator: ICANN (L-root)
- Beyond TLDs and leakage: number of addresses seen

DITL 2017 by the numbers

- About 5.3 TB total
- About 530,000 gzipped pcap files
- About 118 billion lines in those pcaps
- Wide variation in number of files per operator, and also number of lines per file
- Wide variation in how the pcap files are named and the directory structure
- Note: no data from US DOD (G-root)

Step 1: Extract to speed up the later processing

- Read the gzipped pcap files and extract just the source address, RD bit, QTYPE, and QNAME
  - Space delimited for easier parsing later
  - 4.2TB total
- Output file names preserve the root letter but not other directory structure data
- For example, the file “h-0290246” starts with:

```
 24.93.50.9 N A www.twitch.tv.Belkin
2001:1890:1ff:9c8:151:164:110:243 N AAAA EPSON21004D.local
192.221.146.134 N AAAA 247.276.076.173
54.76.186.194 N A auffahrrampen.kaufen
2001:1890:1ff:9c5:151:164:110:139 N AAAA HT-
RT5\032F8491B1.local
217.118.66.96 N A qbezxvktypnhj
```

Step 2: Collect the the TLDs seen

- And here we hit memory limits
- The root servers are often bombarded with requests for randomly-generated strings
- For example, also from file “h-0290246”:

```
83.169.185.35 N AAAA issjycemxgwyrzr
200.49.130.47 N A yccruqifmbcfpo
200.49.130.51 N A axpkbwbmiwfpi
2800:480:ff78:5::2 N A zclsxyhvki
24.29.108.104 N A manlchkioqcl
71.250.0.138 N A yjqbokp
```

Solution: subset and sample

- Don’t try to count everything, but try to make reasonable buckets for the memory-busting categories
- Keep separate the names that are in the root zone, in the RFC 6761 registry, in the top 100 unknown TLDs from the Interisle report, in the gTLD applications, or in the P2P specials Internet Draft
- Split into “has a dot” and “bare TLD”
  - “close.skyworth” and “54.169.175.88:9000” have a dot; “umqfgvujuj” and “unqruefp” are bare TLDs
  - Is the content of the long tail of queries important, or just the size?
- Use a sample of 10% of the files to be a representation of the whole

DITL: all results

Rank	String	Count	Legend
1	tld_no_sub	41036930	
2	com	17559799	I
3	tld_with_sub	12876816	
4	net	8822637	I
5	.	4510507	
6	local	3949823	6I
7	home	3178447	IA
8	org	1614571	I
9	arpa	1318627	I
10	cn	981539	I
11	lan	834007	I
12	z	736604	
13	localdomain	666603	I
14	ru	640350	I
15	dhcp	572190	
16	internal	538401	I
17	uk	525466	I
:	.	.	

DITL: results that are not in the root

Rank	String	Count	Legend
1	tld_no_sub	41036930	
2	tld_with_sub	12876816	
3	local	3949823	6I
4	home	3178447	IA
5	lan	834007	I
6	z	736604	
7	localdomain	666603	I
8	dhcp	572190	
9	internal	538401	I
10	belkin	300765	I
11	dlink	285790	I
12	localhost	284949	6I
13	invalid	273626	6I
14	corp	256302	IA
15	workgroup	197854	
16	homestation	192762	I
17	ip	172206	
:	. .		

A few observations from this DITL data

- Are all of those leaked?
  - We don't know, and probably can't tell for sure
  - It's a weird mix of non-root TLDs seen
- Note that "z" and "dhcp" and "workgroup" and "ip" appear to be new (or much more leaked) since the Interisle report five years ago
- How this might inform future rounds of adding new gTLDs?
  - We got used to saying "home, corp, and mail" but now that might need to change
  - Many of the top 25 from Interisle report have already been delegate with no reports of significant damage

The DNS-OARC system that did this analysis

- Started work on the normal machine, **an1**
- It has very little free disk space (currently 300 GB) and 16 GB RAM, but has 64 cores
- The DITL data is on a different machine, so reading it was over the network
- DNS-OARC provisioned a new machine for this work, **an4**
- 24 cores, 64 GB RAM, 15 TB hard disk
  - Also connected to the same data-holding systems that an1 is
- Being able to keep interim dataset on local disk massively sped things up, even with slower cores

Same data, but just from ICANN's root

- Processed on ICANN's research machines
- Similar, but not completely

L-root: all results

Rank	String	Count	Legend
1	tld_no_sub	2775931	
2	com	999796	I
3	tld_with_sub	667696	
4	net	517211	I
5	.	315357	
6	home	238113	IA
7	local	173965	6I
8	kr	95055	I
9	cn	86775	I
10	dhcp	86086	
11	org	82025	I
12	arpa	58369	I
13	lan	52697	I
14	localdomain	44592	I
15	ip	39570	
16	ru	32281	I
17	openstacklocal	31485	
:	. .		

L-root: results that are not in the root

Rank	String	Count	Legend
1	tld_no_sub	2775931	
2	tld_with_sub	667696	
3	home	238113	IA
4	local	173965	6I
5	dhcp	86086	
6	lan	52697	I
7	localdomain	44592	I
8	ip	39570	
9	openstacklocal	31485	
10	internal	26989	I
11	localhost	26319	6I
12	dlink	22670	I
13	invalid	21996	6I
14	davolink	14666	
15	workgroup	14566	
16	belkin	14479	I
17	gateway	14236	I
:	. .		

A few observations from this L-root data

- Some TLDs are queried for much more often in L-root than in the full DITL data
- “kr” appears before “cn” in the L-root data, but has less than half as many hits in the full DITL data
- “z” is ranked 12 of non-root data in DITL, not even in the top 100 for L-root
- But overall, many of the same names in the same rankings

Number of addresses seen

- Was not the original intent of the study, threw it in near the end
- Can't measure it across the full DITL data because Netnod (I-root) anonymizes it's addresses
  - 10.87.193.237, 10.247.84.0, 10.3.67.118, ...
  - Likely more operators will do so for DITL 2018 and beyond
- There was a question about whether queries sent to the root with RD turned on would make an analysis difference
- The following is just about L-root data, done from our own captures, not the DITL data

L-root on DITL day

```
N:     3274969
R:      249403
All:   3496081
```

L-root around DITL week

Date		Count	Total	Incr.	Incr. %
20170410	N:	3190313	3190313	3190313	100.0%
20170410	R:	238220	238220	238220	100.0%
20170410	All:	3401307	3401307	3401307	100.0%
20170411	N:	3274969	4239972	1049659	24.8%
20170411	R:	249403	386558	148338	38.4%
20170411	All:	3496081	4579700	1178393	25.7%
20170412	N:	3229865	5035805	795833	15.8%
20170412	R:	237562	511291	124733	24.4%
20170412	All:	3439395	5480261	900561	16.4%
.	.	.			

L-root for DITL week

Date		Count	Total	Incr.	Incr. %
20170410	N:	3190313	3190313	3190313	100.0%
20170410	R:	238220	238220	238220	100.0%
20170410	All:	3401307	3401307	3401307	100.0%
20170411	N:	3274969	4239972	1049659	24.8%
20170411	R:	249403	386558	148338	38.4%
20170411	All:	3496081	4579700	1178393	25.7%
.	.	.			
20170416	N:	2779412	7213695	421249	5.8%
20170416	R:	211126	945164	92846	9.8%
20170416	All:	2970472	8019437	499354	6.2%

BUT:

Entries that had only 1 hit during the week:
N: 22.1% R: 24.8% All: 22.1%
Entries that had 7 or fewer hits during the week:
N: 53.0% R: 61.0% All: 53.4%

Questions and more to come

- Software on GitHub
- DITL 2018 is coming soon
- We would love to hear what analysis you would like us to do with our L-root data