Zombies

Geoff Huston
APNiC Labs
What we did:

Run an online advertisement with an embedded measurement script

– The script caused the browser to fetch a number of 1x1 ‘blots’

– To ensure that we had a clear view of the actions of the user and the DNS resolvers they use, we used unique URL labels.
We are currently serving some 8 M Ad impressions per day.
URL Load

• We are generating some 24 million DNS queries for “unique” DNS names per day
• And similarly performing some 24 million HTTP blot fetches for “unique” URLs per day
"Unique"?

What is meant by “unique”?

– The DNS name is queried by a single endpoint once and only once(*) – never again!
  (And the name includes a subfield of the time it was created)

– The TTL of the record is 1 second

– The URL fetch is performed by a single endpoint once and only once – and never again!

– Which means that we should see one query for the name at the authoritative name server

* Well not quite, 25% of the time its queried twice, and sometimes more, but its all triggered by a single resolution action initiated by the endpoint – all these queries are clustered together in time
What do we see?

1450151673.887 [x] 15–Dec–2015 query: z.t10000.u953a6ea5.s1448087430.i5112.vxxxx.06ca0.z.dotnxdomain.net A
1450151673.887 [x] 15–Dec–2015 query: z.t10000.uc86fd1d9.s1447672979.i5112.vxxxx.3b460.z.dotnxdomain.net A
1450151673.887 [x] 15–Dec–2015 query: z.t10000.ub46e3821.s1447703026.i5112.vxxxx.0c914.z.dotnxdomain.net A
1450151674.013 [x] 15–Dec–2015 query: z.t10000.u953a6ea5.s1448087430.i5112.vxxxx.06ca0.z.dotnxdomain.net A
1450151674.015 [x] 15–Dec–2015 query: z.t10000.ub46e3821.s1447703026.i5112.vxxxx.0c914.z.dotnxdomain.net A
1450151674.017 [x] 15–Dec–2015 query: z.t10000.uc86fd1d9.s1447672979.i5112.vxxxx.3b460.z.dotnxdomain.net A
1450151674.753 [x] 15–Dec–2015 query: z.t10000.u953a6ea5.s1448087430.i5112.vxxxx.06ca0.z.dotnxdomain.net A
1450151674.755 [x] 15–Dec–2015 query: z.t10000.uc86fd1d9.s1447672979.i5112.vxxxx.3b460.z.dotnxdomain.net A
1450151674.756 [x] 15–Dec–2015 query: z.t10000.u953a6ea5.s1448087430.i5112.vxxxx.06ca0.z.dotnxdomain.net A
1450151674.757 [x] 15–Dec–2015 query: z.t10000.ub46e3821.s1447703026.i5112.vxxxx.0c914.z.dotnxdomain.net A
What do we see?

The time that the ad was created!

2015-12-15 03:54:33

query time

2015-11-21 06:30:30
2015-11-16 11:22:59
2015-11-16 19:43:46
2015-11-21 06:30:30
2015-11-16 11:22:59
2015-11-21 06:30:30
2015-11-16 19:43:46
What do we see?

Query Time

Creation Time

Diff == Zombie Time!
One Day, One DNS Server
One Day, One DNS Server

![Zombie Age Distribution Graph]

The graph shows the distribution of zombie query counts over the age of query names in days.
60 Days, All DNS Servers
60 Days, All DNS Servers

50% of all zombie queries are more than 6 months old!
Zombie Repeats per day
Zombie Repeats per day

1 query every 3 seconds!
Zombie Repeats per day

Zombies per Day (log scale)

Number of repeat queries per day

Zombie DNS Name Count

Zombie Repeats per day chart.
Zombie Repeats per day

2/3 of all queries occur once per day

Do Zombies have a 32 “hours” per day?
What is causing this?

Is this the result of a collection of deranged DNS recursive resolvers with an obsession about never forgetting a thing?

Or web proxies that just have too much time (and space) on their hands and want to fill all that space with a vast collection of identical 1x1 pixel gifs?

Let’s look at web zombies ...
Zombie URL Age Distribution

![Graph showing the distribution of zombie URLs by age. The x-axis represents the age of the URL zombie in days, while the y-axis shows the number of URL zombies (log scale). The graph demonstrates a rapid decrease in the number of zombie URLs as their age increases, followed by a plateau at around 100 zombie URLs per day.]
50% of all zombie URLs are less than 4 days old
DNS vs URLs

DNS zombies are living their own zombie half life! They are not the hell spawn of zombie URLs!
Zombies

• It seems that on the Internet very little is allowed to be forgotten

• We can use this...
DNS as storage

Write(index, data)
  query = "data.index.storage"
  foreach i (0..100) { dig IN A query; }

DNS as storage

Write(index, data)

query = "data.index.storage"

foreach i (0..100) { dig IN A query; }

Read(index)

wait(query, "index.storage")

return data
DNS as storage

Write(index, data)
query = “data.index.storage”
foreach i (0..100) { dig IN A query; }

Read(index)
wait(query, “index.storage”) return data

Delete(index)
print(“I’m sorry Dave, I can’t do that”)
Thanks!