What's In the Cache?

Ralf Weber, Principal Architect, Special Projects
### Introduction

**About me:**
- Old grey bearded Unix sysadmin
- Ran DNS Caching servers for 15 years
  - Kevag Telekom
  - Colt Telecom
- Now help people deploy Caching servers
  - Nominum
- Obsessed with Cache Hits ;-)  

**About my work (Nominum):**
- We do awesome DNS engines
- We also offer DNS policy enhanced applications
  - Security
  - Parental Controls
- To achieve success in the above we analyze DNS data
  - A subset of our customers DNS data
  - Anonymized client resolver data
  - Statistical data of the server (Telemetry)
  - All statistics of the server are emitted every 5 seconds
  - Includes QPS and Cache Hit Rate
Cache Hit Rate in comparison to qps
Cache Hit Rate in comparison to qps

0.0021 % can be ignored
Zoom into Cache Hit Rate vs. QPS
Cache Hit Rate summarised

- Cache Hit rate is over 90% for normal servers
- Servers that serve <1000 qps are mostly lab servers and total traffic is negligible
- Loaded servers >40k qps have above 95% Cache Hit Rate
Core Domains / Data Set

What is an core domain
- Usually a second level domain
- In some cases (co.uk) deeper
- Delegated domains from Registries
- Based on Mozilla Public Suffix List

Why are we using it
- Looking for a meaningful domain boundaries
- Usually owned by one organization, though one organization can have multiple core domains
- Easy to digest in the server (NS lookups e.g would be harder for xx millions of domains)

Data set
- One day of DNS data from customers around the world
- Only select servers from customers mostly medium qps
- 100 billion queries
- 7 million core domains
- Get used to logarithmic scales
- Ratio A to AAAA = 2/3 to 1/3
Core domain distribution
Core domain distribution

- 67%
- 90%
## Core Domains

<table>
<thead>
<tr>
<th>Domain</th>
<th>%</th>
<th>CHR%</th>
</tr>
</thead>
<tbody>
<tr>
<td>apple.com</td>
<td>7.8</td>
<td>91.2</td>
</tr>
<tr>
<td>google.com</td>
<td>5.9</td>
<td>99.8</td>
</tr>
<tr>
<td>facebook.com</td>
<td>4.4</td>
<td>99.9</td>
</tr>
<tr>
<td>akadns.net</td>
<td>2.9</td>
<td>70.4</td>
</tr>
<tr>
<td>googleapis.com</td>
<td>2.8</td>
<td>99.9</td>
</tr>
<tr>
<td>akamaiedge.net</td>
<td>2.5</td>
<td>98.0</td>
</tr>
<tr>
<td>doubleclick.net</td>
<td>2.0</td>
<td>99.9</td>
</tr>
<tr>
<td>microsoft.com</td>
<td>1.9</td>
<td>80.1</td>
</tr>
<tr>
<td>whatsapp.net</td>
<td>1.7</td>
<td>99.9</td>
</tr>
<tr>
<td>netflix.com</td>
<td>1.6</td>
<td>99.8</td>
</tr>
</tbody>
</table>

- Top 100 core domains make 60% of traffic
- Top 1000 core domains make 90%
- Average hit rate 90% in top 1000
- 6 million core domains have less than 10 queries per day
- Let's look at the Cache Hit Rate distribution
Cache Hit Rate Distribution

Domain Rank based on queries

Cache Hit Rate (%)
Zoom into Cache Hit Rate distribution
<table>
<thead>
<tr>
<th>Rank</th>
<th>Domain</th>
<th>%</th>
<th>CHR</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>akadns.net</td>
<td>2.9</td>
<td>70.4</td>
</tr>
<tr>
<td>8</td>
<td>microsoft.com</td>
<td>1.9</td>
<td>80.1</td>
</tr>
<tr>
<td>23</td>
<td>adnxs.com</td>
<td>0.7</td>
<td>75.7</td>
</tr>
<tr>
<td>27</td>
<td>cloudfront.net</td>
<td>0.6</td>
<td>65.8</td>
</tr>
<tr>
<td>33</td>
<td>live.com</td>
<td>0.5</td>
<td>83.0</td>
</tr>
<tr>
<td>80</td>
<td>windows.com</td>
<td>0.2</td>
<td>80.0</td>
</tr>
<tr>
<td>86</td>
<td>dyndns.org</td>
<td>0.1</td>
<td>79.1</td>
</tr>
<tr>
<td>92</td>
<td>akamaihd.net</td>
<td>0.1</td>
<td>73.0</td>
</tr>
</tbody>
</table>

- Most popular domains have a good cache hit rate (>95%)
- Some cache hit rate seem low
  - More on that later
- Overall average Cache Hit Rate:
  - 91.46
- So lets look into DNSSEC
DNSSEC Cache Hit Rate

![Graph showing DNSSEC Cache Hit Rate against Domain Rank based on queries. The graph displays a scatter plot with cache hit rates on the Y-axis and domain rank on the X-axis. The data points are spread across the graph, indicating a trend in the cache hit rate as the domain rank increases.](image)
## DNSSEC

<table>
<thead>
<tr>
<th>#</th>
<th>Domain</th>
<th>%</th>
<th>CHR</th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>vkcache.com</td>
<td>0.36</td>
<td>99.7</td>
</tr>
<tr>
<td>48</td>
<td>nist.gov</td>
<td>0.32</td>
<td>99.8</td>
</tr>
<tr>
<td>70</td>
<td>vmware.com</td>
<td>0.18</td>
<td>98.9</td>
</tr>
<tr>
<td>77</td>
<td>gmx.net</td>
<td>0.16</td>
<td>98.9</td>
</tr>
<tr>
<td>118</td>
<td>cpsc.gov</td>
<td>0.11</td>
<td>99.9</td>
</tr>
<tr>
<td>154</td>
<td>ioam.de</td>
<td>0.08</td>
<td>99.9</td>
</tr>
<tr>
<td>188</td>
<td>pki.goog</td>
<td>0.06</td>
<td>99.9</td>
</tr>
<tr>
<td>202</td>
<td>cloudflare.com</td>
<td>0.06</td>
<td>99.2</td>
</tr>
<tr>
<td>203</td>
<td>mozilla.org</td>
<td>0.06</td>
<td>96.1</td>
</tr>
</tbody>
</table>

- Not much DNSSEC domains
  - Total 5746 (0.0%)  
  - Total traffic validatable is 2.5 %  
- Cache Hit Rate
  - Very good at popular domains  
  - Long tail falls off  
  - Average is 33%  
- Lets look at IPv6
  - No real user traffic on v6  
  - But A to AAAA distribution
Core Domain Distribution A vs AAAA
Cache Hit Rate A vs AAA
# Core Domains A vs AAAA

<table>
<thead>
<tr>
<th>Domain (A)</th>
<th>%</th>
<th>CHR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>apple.com</td>
<td>7.8</td>
<td>99.7</td>
</tr>
<tr>
<td>google.com</td>
<td>6.1</td>
<td>99.8</td>
</tr>
<tr>
<td>facebook.com</td>
<td>4.5</td>
<td>99.9</td>
</tr>
<tr>
<td>googleapis.com</td>
<td>2.9</td>
<td>99.9</td>
</tr>
<tr>
<td>akamaiedge.net</td>
<td>1.7</td>
<td>97.6</td>
</tr>
<tr>
<td>akadns.net</td>
<td>1.6</td>
<td>99.2</td>
</tr>
<tr>
<td>netflix.com</td>
<td>1.4</td>
<td>99.8</td>
</tr>
<tr>
<td>doubleclick.net</td>
<td>1.4</td>
<td>99.9</td>
</tr>
<tr>
<td>microsoft.com</td>
<td>1.2</td>
<td>98.0</td>
</tr>
<tr>
<td>whatsapp.net</td>
<td>1.2</td>
<td>99.9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Domain (AAAA)</th>
<th>%</th>
<th>CHR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>apple.com</td>
<td>8.2</td>
<td>72.7</td>
</tr>
<tr>
<td>google.com</td>
<td>5.8</td>
<td>99.7</td>
</tr>
<tr>
<td>facebook.com</td>
<td>4.3</td>
<td>99.9</td>
</tr>
<tr>
<td>akadns.net</td>
<td>4.2</td>
<td>32.6</td>
</tr>
<tr>
<td>googleapis.com</td>
<td>2.9</td>
<td>99.9</td>
</tr>
<tr>
<td>akamaiedge.net</td>
<td>2.4</td>
<td>98.8</td>
</tr>
<tr>
<td>doubleclick.net</td>
<td>2.3</td>
<td>99.9</td>
</tr>
<tr>
<td>microsoft.com</td>
<td>2.1</td>
<td>45.4</td>
</tr>
<tr>
<td>amazonaws.com</td>
<td>1.7</td>
<td>92.4</td>
</tr>
<tr>
<td>whatsapp.net</td>
<td>1.5</td>
<td>99.8</td>
</tr>
</tbody>
</table>
# IPv6 (AAAA) Cache Hit Rate

<table>
<thead>
<tr>
<th>#</th>
<th>Domain</th>
<th>%</th>
<th>CHR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>apple.com</td>
<td>8.2</td>
<td>72.7</td>
</tr>
<tr>
<td>4</td>
<td>akadns.net</td>
<td>4.3</td>
<td>32.6</td>
</tr>
<tr>
<td>8</td>
<td>microsoft.com</td>
<td>2.1</td>
<td>45.2</td>
</tr>
<tr>
<td>24</td>
<td>live.com</td>
<td>0.6</td>
<td>15.4</td>
</tr>
<tr>
<td>44</td>
<td>skype.com</td>
<td>0.3</td>
<td>53.6</td>
</tr>
<tr>
<td>66</td>
<td>windows.com</td>
<td>0.2</td>
<td>49.0</td>
</tr>
<tr>
<td>89</td>
<td>mcafee.com</td>
<td>0.1</td>
<td>12.0</td>
</tr>
</tbody>
</table>

- Some domains especially in the top domains have absurdly low cache hit rate
- In general the IPv6 rates in the long tail are a bit lower, but nothing to what we see in the top
- Overall hit rate comparison
  - Top 100 A=97 AAAA=91
  - Top 1000 A=91 AAAA=87
- Overall traffic rate is 2/3 A and 1/3 AAAA
- So is IPv6 hindered by records expiring out of the cache
- Lets look into the data....
Normal query

; <<>> DiG 9.11.0-P3 <<>> +norec +nocookie dns-oarc.net @ns.dns-oarc.net. A; <<>> DiG 9.11.0-P3 <<>> +norec +nocookie dns-oarc.net @ns.dns-oarc.net. AAAA

;; global options: +cmd
;; Got answer:
;; -->HEADER<<- opcode: QUERY, status: NOERROR, id: 24774
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
dns-oarc.net. IN A

;; ANSWER SECTION:
dns-oarc.net. 3600 IN A 64.191.0.198

;; Query time: 5 msec
;; SERVER: 2620:ff:c000:0:1::65#53(2620:ff:c000:0:1::65)
;; WHEN: Fri Sep 29 16:38:37 PDT 2017
;; MSG SIZE  rcvd: 57

; <<>> DiG 9.11.0-P3 <<>> +norec +nocookie dns-oarc.net

;; global options: +cmd
;; Got answer:
;; -->HEADER<<- opcode: QUERY, status: NOERROR, id: 15590
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
dns-oarc.net. IN AAAA

;; ANSWER SECTION:
dns-oarc.net. 3600 IN AAAA 2620:ff:c000::198

;; Query time: 7 msec
;; SERVER: 2620:ff:c000:0:1::65#53(2620:ff:c000:0:1::65)
;; WHEN: Fri Sep 29 16:50:32 PDT 2017
;; MSG SIZE  rcvd: 69
No AAAA exist query

; <<>> DiG 9.11.0-P3 <<>> slashdot.org A
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 3584
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 5, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;slashdot.org. IN A

;; ANSWER SECTION:
slashdot.org. 300 IN A 216.34.181.45

;; AUTHORITY SECTION:
slashdot.org. 86400 IN NS ns3.dnsmadeeasy.com.
slashdot.org. 86400 IN NS ns4.dnsmadeeasy.com.
slashdot.org. 86400 IN NS ns0.dnsmadeeasy.com.
slashdot.org. 86400 IN NS ns1.dnsmadeeasy.com.
slashdot.org. 86400 IN NS ns2.dnsmadeeasy.com.

;; Query time: 21 msec
;; SERVER: 2620:0:ce0:100::12#53(2620:0:ce0:100::12)
;; WHEN: Fri Sep 29 16:37:06 PDT 2017
;; MSG SIZE  rcvd: 162

; <<>> DiG 9.11.0-P3 <<>> slashdot.org AAAA +multiline
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 334
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;slashdot.org. IN AAAA

;; AUTHORITY SECTION:
slashdot.org. 300 IN SOA ns0.dnsmadeeasy.com.
hostmaster.slashdotmedia.com. (2016045443 ; serial
14400 ; refresh (4 hours)
600 ; retry (10 minutes)
604800 ; expire (1 week)
300 ; minimum (5 minutes)
)

;; Query time: 25 msec
;; SERVER: 2620:0:ce0:100::12#53(2620:0:ce0:100::12)
;; WHEN: Fri Sep 29 17:02:31 PDT 2017
;; MSG SIZE  rcvd: 121
5 - Caching Negative Answers

Like normal answers negative answers have a time to live (TTL). As there is no record in the answer section to which this TTL can be applied, the TTL must be carried by another method. This is done by including the SOA record from the zone in the authority section of the reply. When the authoritative server creates this record its TTL is taken from the minimum of the SOA.MINIMUM field and SOA's TTL. This TTL decrements in a similar manner to a normal cached answer and upon reaching zero (0) indicates the cached negative answer MUST NOT be used again.
Akadns

; <<>> DiG 9.11.0-P3 <<>> gsp64-ssl.ls-apple.com.akadns.net.
; global options: +cmd
; Got answer:
; -->HEADER<< opcode: QUERY, status: NOERROR, id: 26202
; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; QUESTION SECTION:
gsp64-ssl.ls-apple.com.akadns.net. IN A

;; ANSWER SECTION:
gsp64-ssl.ls-apple.com.akadns.net. 234 IN A 17.142.169.200
  gsp64-ssl.ls-apple.com.akadns.net. 234 IN A 17.142.169.199

;; Query time: 25 msec
;; SERVER: 2620:0:ce0:100::12#53(2620:0:ce0:100::12)
;; WHEN: Fri Sep 29 17:08:03 PDT 2017
;; MSG SIZE  rcvd: 94

; <<>> DiG 9.11.0-P3 <<>> gsp64-ssl.ls-apple.com.akadns.net.
AAAA
; global options: +cmd
; Got answer:
; -->HEADER<< opcode: QUERY, status: NOERROR, id: 30864
; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; QUESTION SECTION:
gsp64-ssl.ls-apple.com.akadns.net. IN AAAA

;; Query time: 42 msec
;; SERVER: 2620:0:ce0:100::12#53(2620:0:ce0:100::12)
;; WHEN: Fri Sep 29 17:08:23 PDT 2017
;; MSG SIZE  rcvd: 62
Negative responses without SOA records SHOULD NOT be cached as there is no way to prevent the negative responses looping forever between a pair of servers even with a short TTL.
;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;cid-24e99ee8e7b6bdf0.users.storage.live.com. IN A

;; ANSWER SECTION:
cid-24e99ee8e7b6bdf0.users.storage.live.com. 600 IN A 157.55.109.224

;; Query time: 54 msec
;; SERVER: 134.170.104.152#53(134.170.104.152)
;; WHEN: Fri Sep 29 17:27:38 PDT 2017
;; MSG SIZE rcvd: 88

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;cid-24e99ee8e7b6bdf0.users.storage.live.com. IN AAAA

;; AUTHORITY SECTION:
storage.live.com. 0 IN SOA geodns.storage.skyprod.akadns.net. msnhst.microsoft.com. (2007010101 ; serial 43200 ; refresh (12 hours) 21600 ; retry (6 hours) 86400 ; expire (1 day) 0 ; minimum (0 seconds) )

;; Query time: 58 msec
;; SERVER: 134.170.104.152#53(134.170.104.152)
;; WHEN: Fri Sep 29 17:29:52 PDT 2017
;; MSG SIZE rcvd: 158
What???

- This is clearly bad behaviour
  - A lot of domains don’t have AAAA records yet
  - I doubt that it does matter to bring them online in seconds when they get them
  - Defeats the purpose of negative caching
- However my analysis was based on client query data without answers, so can’t quantify how bad
- Unfortunately this data only exists in the recursor to authoritative data which is hard to correlate with the client data
- So lets dig into that data without correlation
  - \( V_4 \) = queries that asked for A
  - \( V_6 \) = queries that asked for AAAA
  - \( V_{4no} \) = NoAnswer/NoError for A
  - \( V_{6no} \) = NoAnswer/NoError for AAAA
  - \( V_{4nc} \) = Non cacheable NoAnswer/NoError for A
  - \( V_{6nc} \) = Non cacheable NoAnswer/NoError for AAAA
Trying to quantify the problem

<table>
<thead>
<tr>
<th>Domain</th>
<th>CHRv6</th>
<th>v4</th>
<th>v6</th>
<th>v4no</th>
<th>v4nc</th>
<th>v6no</th>
<th>v6nc</th>
</tr>
</thead>
<tbody>
<tr>
<td>apple.com</td>
<td>72.7</td>
<td>67.0</td>
<td>32.4</td>
<td>7.9</td>
<td>1.1</td>
<td>53.6</td>
<td>0.8</td>
</tr>
<tr>
<td>akadns.net</td>
<td>32.6</td>
<td>5.5</td>
<td>94.5</td>
<td>0.0</td>
<td>0.3</td>
<td>0.0</td>
<td>97.8</td>
</tr>
<tr>
<td>microsoft.com</td>
<td>45.2</td>
<td>20.5</td>
<td>79.3</td>
<td>9.9</td>
<td>0.8</td>
<td>2.0</td>
<td>95.2</td>
</tr>
<tr>
<td>live.com</td>
<td>15.4</td>
<td>62.7</td>
<td>37.2</td>
<td>0.2</td>
<td>0.7</td>
<td>0.2</td>
<td>95.9</td>
</tr>
<tr>
<td>skype.com</td>
<td>53.6</td>
<td>73.9</td>
<td>21.6</td>
<td>7.0</td>
<td>0.5</td>
<td>5.0</td>
<td>0.1</td>
</tr>
<tr>
<td>windows.com</td>
<td>49.0</td>
<td>59.1</td>
<td>40.7</td>
<td>0.1</td>
<td>0.3</td>
<td>98.2</td>
<td>0.2</td>
</tr>
<tr>
<td>mcafee.com</td>
<td>12.0</td>
<td>46.7</td>
<td>51.7</td>
<td>70.1</td>
<td>5.5</td>
<td>2.3</td>
<td>97.2</td>
</tr>
</tbody>
</table>
Conclusion

- Caching works
  - Even in IPv6 stuff is served from Cache
  - Unless authorities decide not to cache
- Rather then work on protocol extensions for less then 10% (for getting more than one record at a time) we should:
  - Work on DNSSEC deployment (get more domains signed)
  - Work on getting current standards deployed as intended
    - Negative Caching
    - Empty non terminals
    - EDNS0

- Questions???