DNS Caching: Running on Zero

Saleem Bhatti
School of Computer Science
University of St Andrews
Using DNS for networking

• (From ILNP – a work in progress.)
• Use of names and name resolution for:
  – Mobility (host and network)
  – Multi-homing and fail-over
  – TE options
  – Load balancing
  – VM management/migration
  – Others ...
• Enhanced use of DNS compared to today:
  – Dynamic updates of DNS records
  – Cache times (TTL) of records may need to be reduced
(Non-)Effectiveness of DNS caching


• DNS caching has reduced effectiveness for edge sites:
  – trace-driven emulation (no experiments)
  – A records could have low TTL (e.g. below 1000s)
  – such low TTL would have low impact on DNS load
DNS experiments at StA [1]

• Experiments in Q4/2009
• Modify TTL values of records in operational DNS server at School of CS, St Andrews
  – 4 DNS servers: Windows ActiveDirectory
  – ~400 DNS clients: Windows, Linux, MacOSX, BSD
• TTL values for successive 7-day periods during normal semester:
  – changed DNS TTL on ActiveDirectory
  – TTL values used: 1800s, 30s, 15s, 0s
• Configured clients not to cache.
DNS experiments at StA [2]

- Passive collection of packets via port mirror:
  - `tcpdump(8)` targeting port 53
  - Captured all DNS packets
- Results shown on following slides are for:
  - A record requests for servers only during the capture period (relevant to ILNP, and less ‘noisy’ data)
  - using 1 second buckets
- Basic statistics:
  - on time-domain data
- Spectral analysis:
  - examination of request rates
- Analysis: home-brew python scripts, NumPy package
## 2009: Basic dataset meta-data

<table>
<thead>
<tr>
<th>Data set name</th>
<th>TTL [s]</th>
<th>Duration [s]$^1$</th>
<th>Total DNS packets captured$^2$</th>
<th>Number of A record requests for 67 servers$^3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>dns1800</td>
<td>1800</td>
<td>601,200</td>
<td>41,868,522</td>
<td>2,004,133</td>
</tr>
<tr>
<td>dns30</td>
<td>30</td>
<td>601,200</td>
<td>71,105,247</td>
<td>2,648,796</td>
</tr>
<tr>
<td>dn15</td>
<td>15</td>
<td>601,200</td>
<td>56,472,027</td>
<td>3,240,675</td>
</tr>
<tr>
<td>dns0</td>
<td>0</td>
<td>601,200</td>
<td>55,868,573</td>
<td>4,501,590</td>
</tr>
</tbody>
</table>

$^1$ from tcpdump timestamps, rounded to nearest second, 7 days = 604,800 seconds, less 3600s temporal guard band for TTL value changes = 601,200 seconds

$^2$ includes all request and response packets to/from port 53 (TCP and UDP), including erroneous requests, retransmissions etc

$^3$ servers that were active during the 4 weeks of data capture
dns1800: A record requests TTL=1800s

DNS A record queries, dns2009-1800

Mean: 3.33 request/s
Std Dev: 3.47 requests/s
Max: 183 requests/s
dns30: A record requests TTL=30s

Mean: 4.41 request/s
Std Dev: 4.27 requests/s
Max: 261 requests/s
dns15: A record requests TTL=15s

Mean: 5.39 request/s
Std Dev: 4.85 requests/s
Max: 123 requests/s
dns0: A record requests TTL=0s

Mean: 7.49 request/s
Std Dev: 4.93 requests/s
Max: 3.69 requests/s
2009 Summary of basic statistics

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>dns1800</td>
<td>3.33</td>
<td>3</td>
<td>3.47</td>
<td>183</td>
</tr>
<tr>
<td>dns30</td>
<td>4.41</td>
<td>4</td>
<td>4.27</td>
<td>261</td>
</tr>
<tr>
<td>dns15</td>
<td>5.39</td>
<td>4</td>
<td>4.85</td>
<td>123</td>
</tr>
<tr>
<td>dns0</td>
<td>7.49</td>
<td>7</td>
<td>4.93</td>
<td>369</td>
</tr>
</tbody>
</table>

60x drop in TTL values results in $\frac{1}{3}$x increase in A record requests. 0 TTL gives (only) 2$\frac{3}{4}$x increase.
2009 Basic spectral analysis

• Create approximate periodogram by counting occurrences of bucket sizes:
  – have used 1s bucket
  – so size of bucket is number of requests/s

• Comparison of periodogram:
  – shows changing dynamics of request rates
  – gives a better view of the trends in request rates
2009 periodograms: 1800s ...
... 30s, 15s, 0s
External – 30s, 15s, 0s

7-day DNS A record query rates, dns2009-0030-o

7-day CDF for DNS A record query rates, dns2009-0000-o

7-day DNS A record query rates, dns2009-0015-o

7-day CDF for DNS A record query rates, dns2009-0015-o

7-day DNS A record query rates, dns2009-0000-o

7-day CDF for DNS A record query rates, dns2009-0000-o
Internal – 30s, 15s, 0s

7-day DNS A record query rates, dns2009-0030-i

7-day CDF for DNS A record query rates, dns2009-0030-i

7-day DNS A record query rates, dns2009-0015-i

7-day CDF for DNS A record query rates, dns2009-0015-i

7-day DNS A record query rates, dns2009-0000-i

7-day CDF for DNS A record query rates, dns2009-0000-i
Who would set DNS TTLs so low?

• Real A record values for some services:
  – TTL = 60 seconds: yahoo
  – TTL = 20 seconds: akamai
  – TTL = 0 seconds: St Andrews, Computer Science

• Note that a site would NOT set low TTLs for:
  – Its own NS records, which identify its DNS servers.
  – The A records related to its NS records.
  – A, CNAME, PTR records for services, e.g. email MX
  – A (mobile) site can make remote some or all of its authoritative DNS servers; some sites do so today.
Acknowledgements

• Thanks to:
  – Stuart Cheshire (Apple)
  – Dave Thaler (Microsoft)
  for information on OS-specific features of DNS operation in end-hosts

• **A Very Big Thanks to:**
  – the **Systems Admin Group at cs.st-andrews.ac.uk**
    for implementing DNS TTL changes
Summary and Conclusion

• Summary:
  – Zero TTL values for edge-site DNS records possible
  – DNS load with zero DNS TTLs seems manageable
  – (Indeed, 1s is good, perhaps better than zero ...)

• Conclusion:
  – Frequent DNS access for records with very low TTL seems practical

• Future work:
  – Scale experiment: analyses of larger DNS deployments
  – Impact of the use of Secure DNS Dynamic Update and cryptographic authentication of DNS look-ups