
Cheltenham
Research

# Reducing DNS Caching
# or
# "How low can we go?"

**Saleem Bhatti**
**School of Computer Science**
**University of St Andrews**

Randall Atkinson
Cheltenham Research

# Motivation for examining DNS

| Layer | IP | ILNP |
|---|---|---|
| Application | FQDN or IP address | FQDN |
| Transport | IP address + port no. | Identifier + port no. |
| Network | IP address | Locator |

- ILNP: cleaner semantics for naming in the IP stack:
  - Assumes consistent use of DNS names in applications
  - Uses DNS for mobility, multi-homing, traffic engineering …
  - http://ilnp.cs.st-andrews.ac.uk/
- So, DNS performance is important for ILNP
- **But this talk is not about ILNP** …

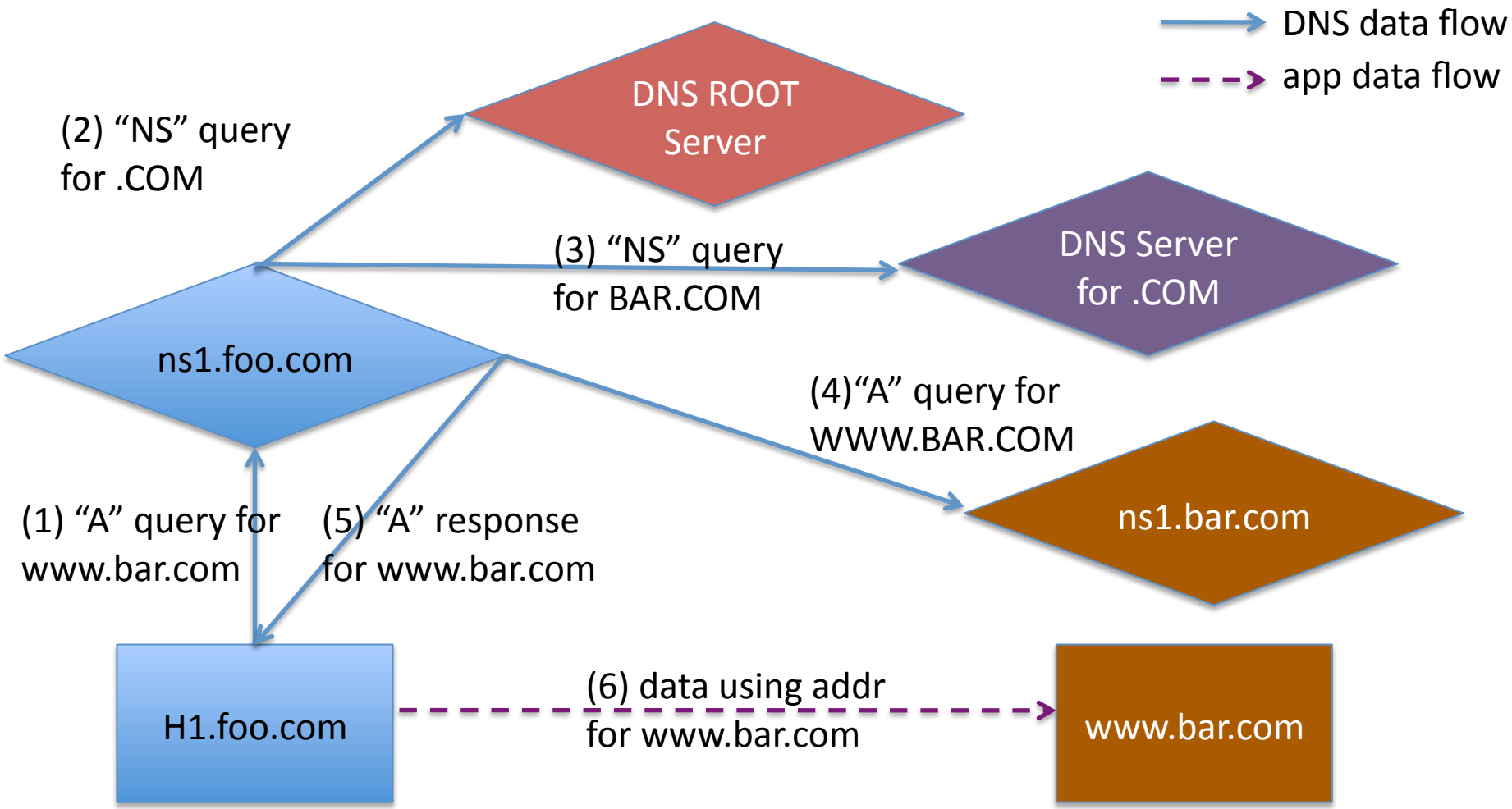
# Domain Name Service

- Distributed name resolution service
- Maps **Fully Qualified Domain Names (FQDNs)** to **DNS records**:
  e.g. FQDN (www.cs.st-andrews.ac.uk) to a DNS **A record** (IPv4 Address, 138.251.206.45),
- Also provides other administrative data for specific services (a simple directory service):
  - Name Servers for a domain (NS records)
  - Mail servers (MX records)
  - Jabber (and other) servers (SRV records)
  - Other record types are possible …

# DNS System Architecture

- Globally distributed name space
- Globally distributed name servers each holding mappings for part of the name space
- Traditionally, read-only for end users
- Enhancements now widely available to enable write access for end users:
  - Secure DNS Dynamic Update (RFC-3007)
  - DNS Security (RFC-4033 to 4035) also useful
  - Implemented in BIND, MS Windows, MS Server

# DNS Lookup Sequence



# Dynamic write access to DNS

- Write access is now available for end users
- There is a temporal caching hierarchy across the spatial distribution of names:
  - Different records get cached for different periods of time, e.g. NS records and A records
  - Maximum caching time defined by a Time To Live (TTL) value held in each DNS record.
- ***Could these two features be exploited in some sensible ways?***

# (Non-)Effectiveness of DNS caching

- Jung, J., Sit, E., Balakrishnan, H., and Morris, R. 2002. *DNS performance and the effectiveness of caching*. IEEE/ACM Trans. on Networking. Vol. 10, No. 5 (Oct. 2002), pp. 589-603.
- DNS caching is ineffective 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
  - used TTL values 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

(awaiting verification)

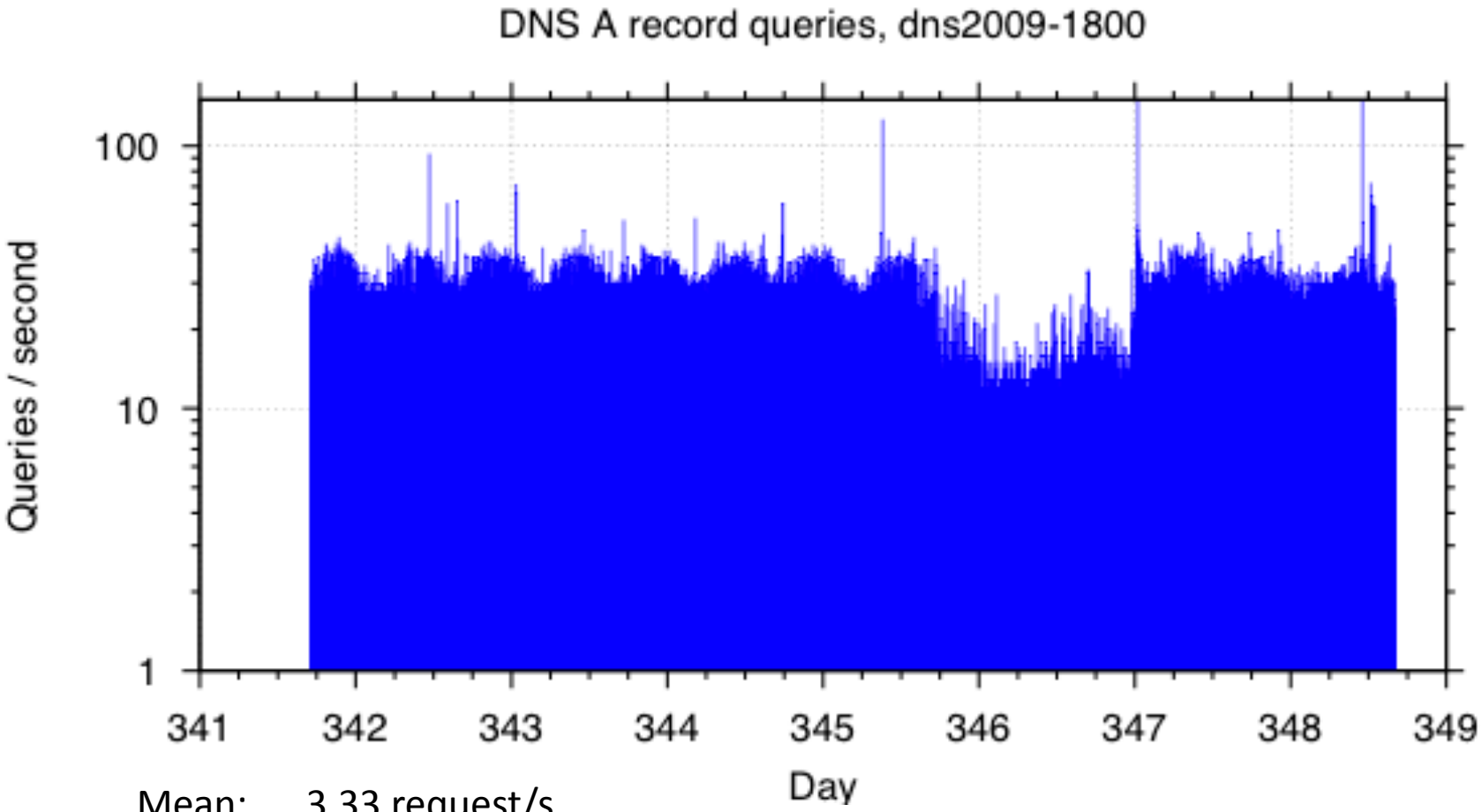
| Data set name | TTL [s] | Duration [s][1] | Total DNS packets captured[2] | Number of A record requests for 67 servers[3] |
|---|---|---|---|---|
| dns1800 | 1800 | 601,200 | 41,868,522 | 2,004,133 |
| dns30 | 30 | 601,200 | 71,105,247 | 2,648,796 |
| dn15 | 15 | 601,200 | 56,472,027 | 3,240,675 |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

[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 etc

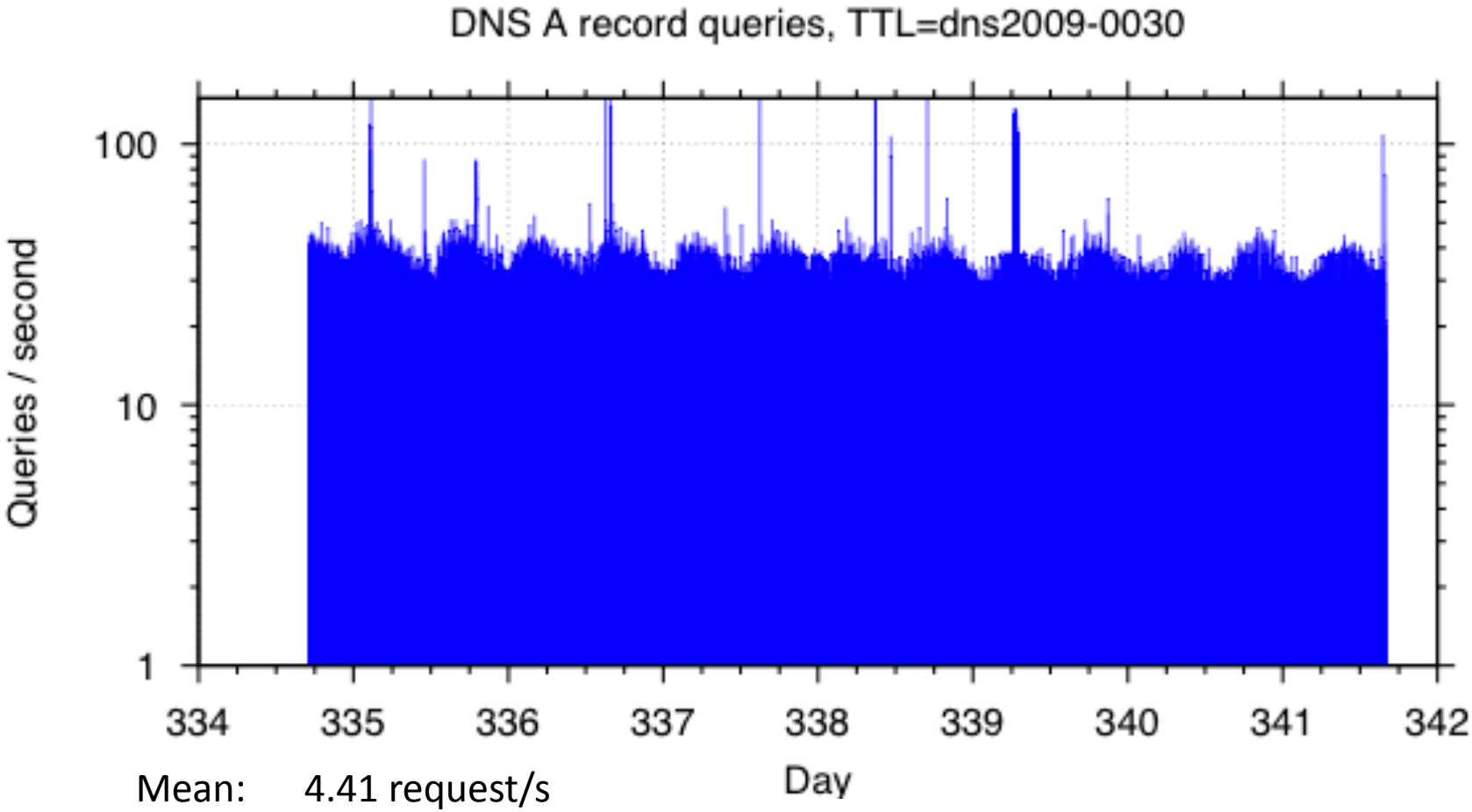
[3] servers that were active during the 4 weeks of data capture

# dns1800: A record requests TTL=1800s



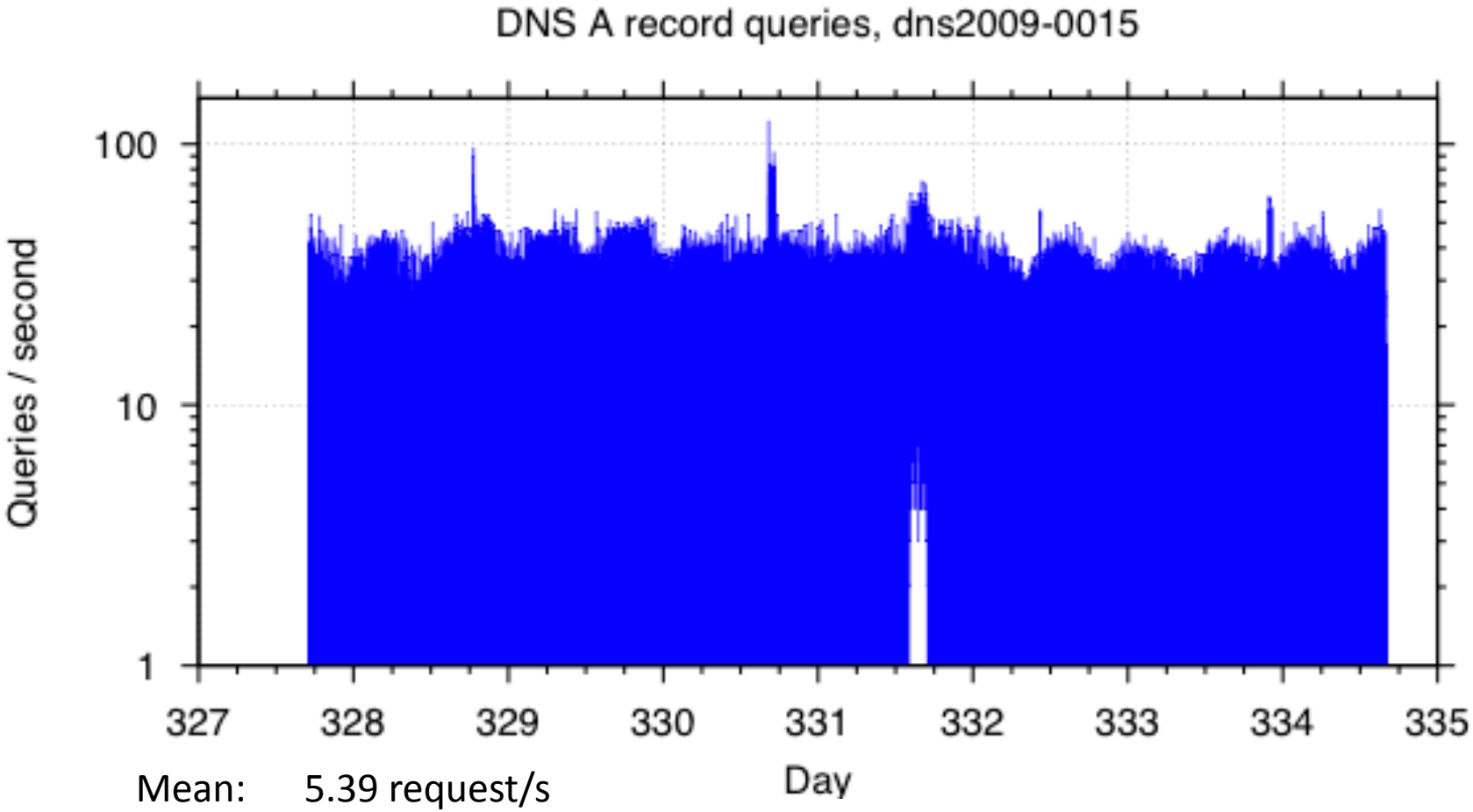

Mean: 3.33 request/s
Std Dev: 3.47 requests/s
Max: 183 requests/s

# dns30: A record requests TTL=30s



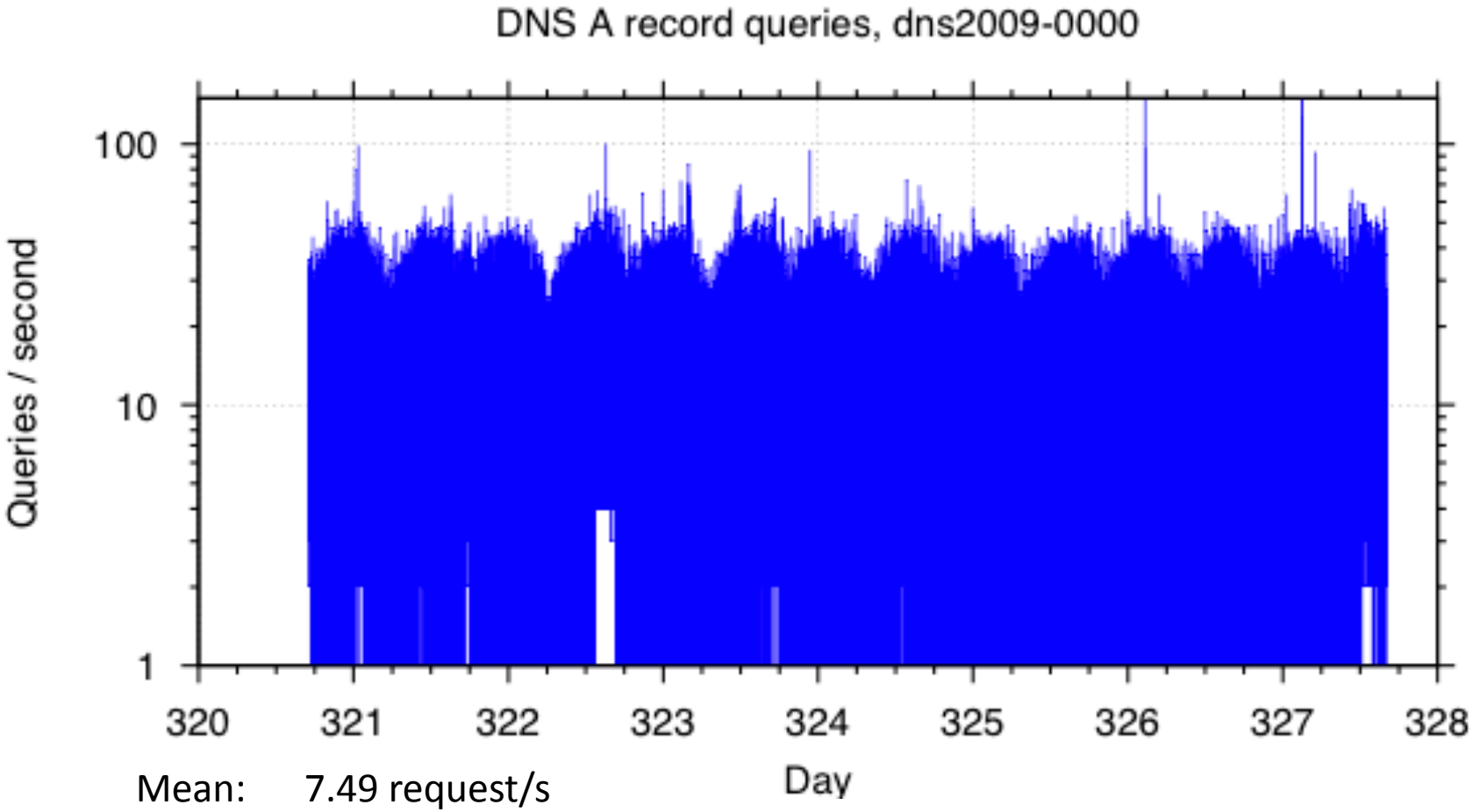

Mean: 4.41 request/s
Std Dev: 4.27 requests/s
Max: 261 requests/s

# 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

(awaiting verification)

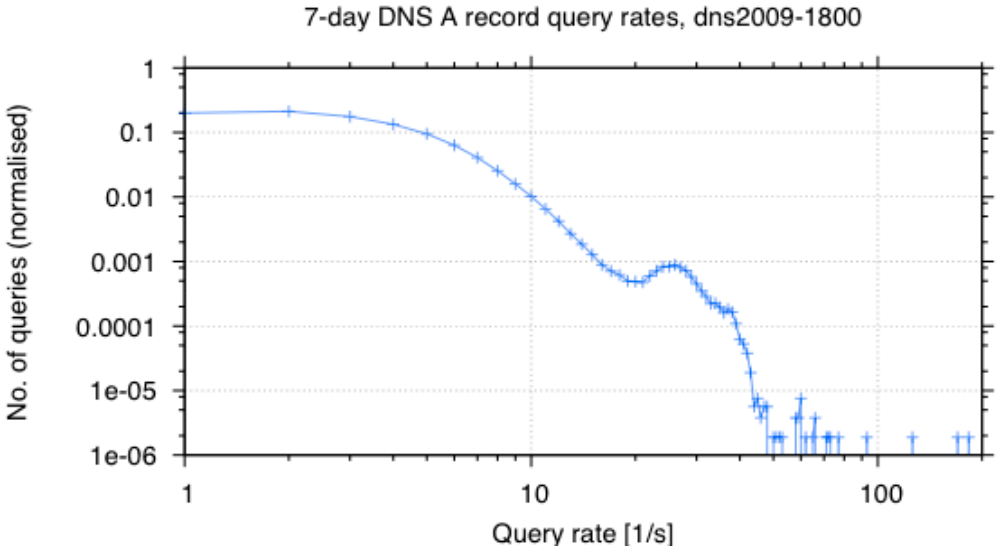
| Data set name | Mean [reqs/s] | Median [reqs/s] | Std Dev [reqs/s] | Maximum [reqs/s] |
|---|---|---|---|---|
| dns1800 | 3.33 | 3 | 3.47 | 183 |
| dns30 | 4.41 | 4 | 4.27 | 261 |
| dns15 | 5.39 | 4 | 4.85 | 123 |
| dns0 | 7.49 | 7 | 4.93 | 369 |

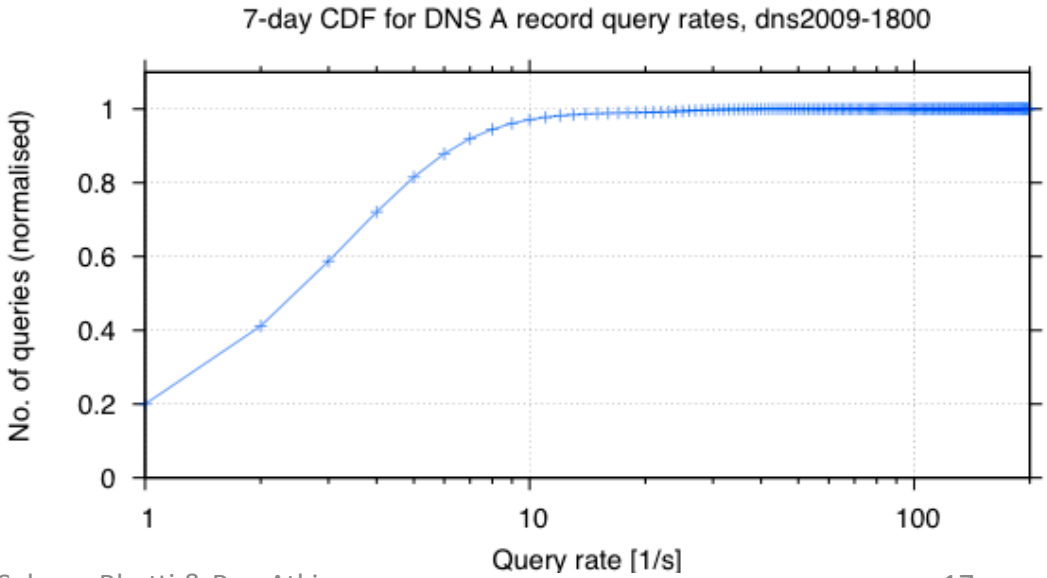
**60x drop** in TTL values results in **⅓x increase** in A record requests: **0 TTL** gives (only) **2¼x increase.**

# 2009 Basic spectral analysis

- Create periodogram by counting frequency 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 understanding of the trends in rates

# 2009 dns1800 perdiogram

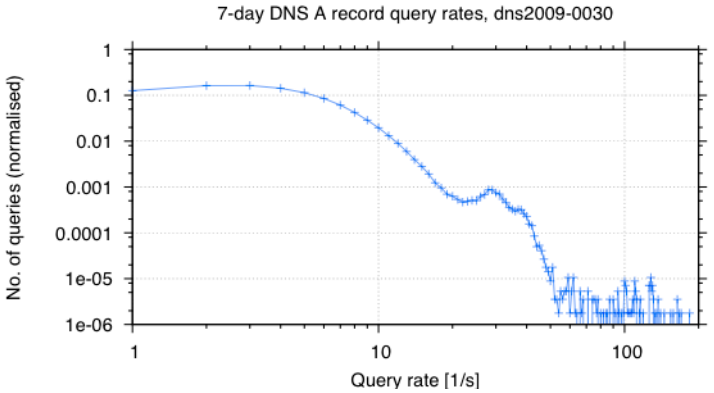


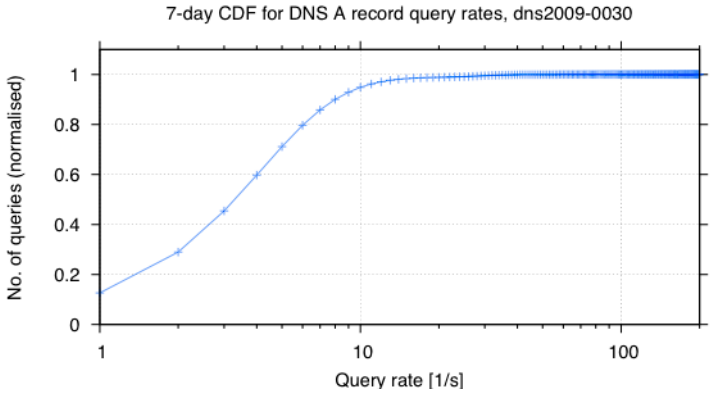



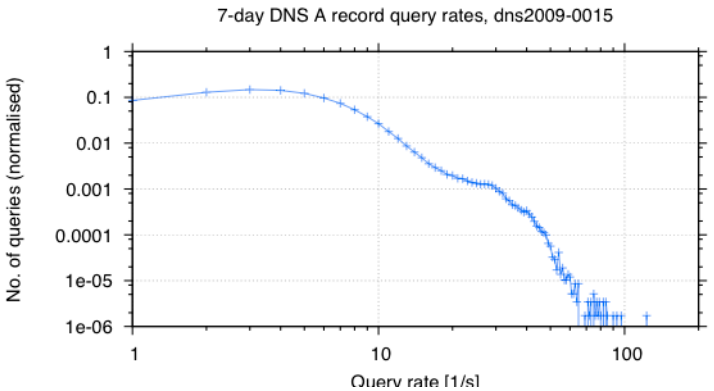
University of St Andrews

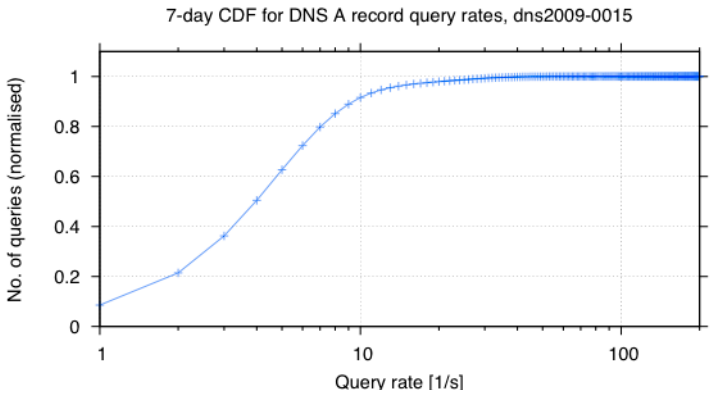
Cheltenham Research

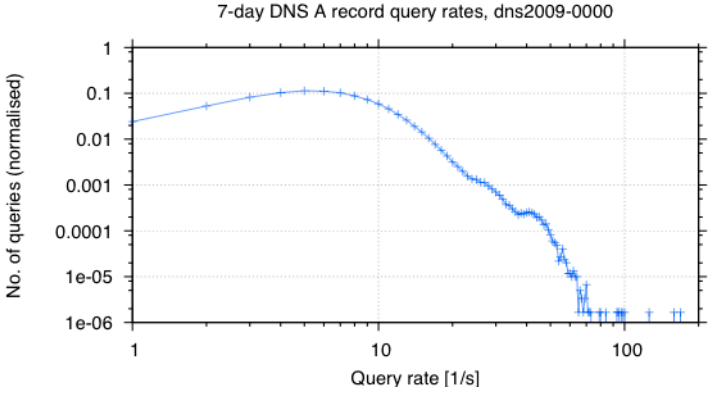
# 2009 dns30, dns15, dns0 perdiograms

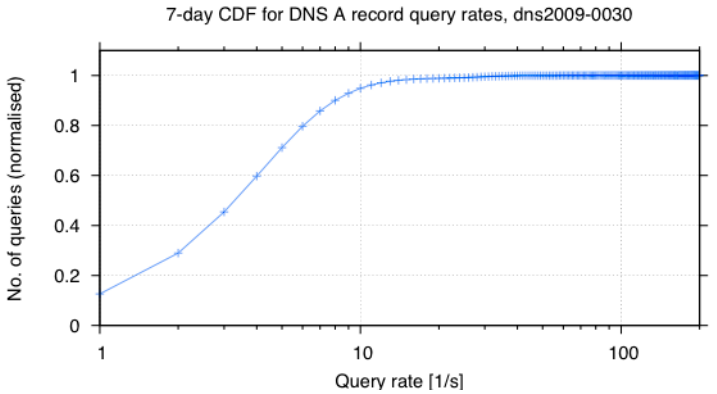


So far, our discussion presents a technical (academic) curiosity.

The interesting question really is:
**“So, what?”**

# What is possible if DNS TTL were zero?

- Dynamic, frequent, & authenticated DNS updates possible:
  - Simulated by Pappas, Hailes, & Giaffreda, published in LCS 2002
  - Very useful for mobility/multi-homing aspects of ILNP
- Help defend against certain network attacks:
  - DNS cache poisoning for end-sites (that do not use DNSSEC)
  - DDoS: fast-cycle multi-homing (i.e. a kind of "fast-flux" DNS for defence rather than attack)
  - Others possible …
- Support for mobility and multi-homing:
  - Location updates give changes in connectivity
- Potential for edge-site based multi-path and TE control:
  - multiple Locator values and DNS L record preferences
- High-speed load balancing and VM mgmt for data centres

# Who would set DNS TTLs so low?

- Real A record values for some servers:
  - TTL = 60 seconds: www.yahoo.com
  - TTL = 20 seconds: content servers in akamai.net
  - **TTL = 0 seconds**: www.cs.st-andrews.ac.uk
- 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.

# Operational Considerations

- Implied semantics of TTL value:
  - **gotcha**: some systems assume that, if network outage time > TTL, then service is down
- PAM2003, PAM2004, NETTS2004 papers by Wessels *et al.*:
  - http://dns.measurement-factory.com/writings/
- In fact, the main site has some very interesting reading, including:
  - http://dns.measurement-factory.com/surveys/200810.html

# Ongoing DNS experiments at StA

- More measurements in 2010/2011:
  - Secure Dynamic DNS updates
- Assess feasibility of supporting ILNP:
  - mobility
  - multi-homing
  - traffic engineering
- Q4/2009 and Q1/2008 data:
  - lots of interesting “features” and behaviour visible in DNS data
  - analysis in progress …

# Summary and Conclusion

- Summary:
  - Zero TTL values for edge-site DNS records possible
  - DNS load with zero DNS TTLs seems manageable
- Conclusion:
  - Both frequent DNS accesses and frequent Dynamic DNS Updates seem practical to deploy
- **A Very Big Thanks to:**
  - **Systems Group at cs.st-andrews.ac.uk for implementing DNS TTL changes**