Handle with Care: Naming, Layering and Caching "Broken" on the Internet

Saleem Bhatti
University of St Andrews
http://www.cs.st-andrews.ac.uk/~saleem/

Outline

- Today's architecture:
  - use of naming, layering and caching today.
- Problems:
  - the way that n/l/c are (mis)used today.
- Impact:
  - why this is "broken" and the impact it will have.
- No solutions in this talk … only questions!

Naming

Names

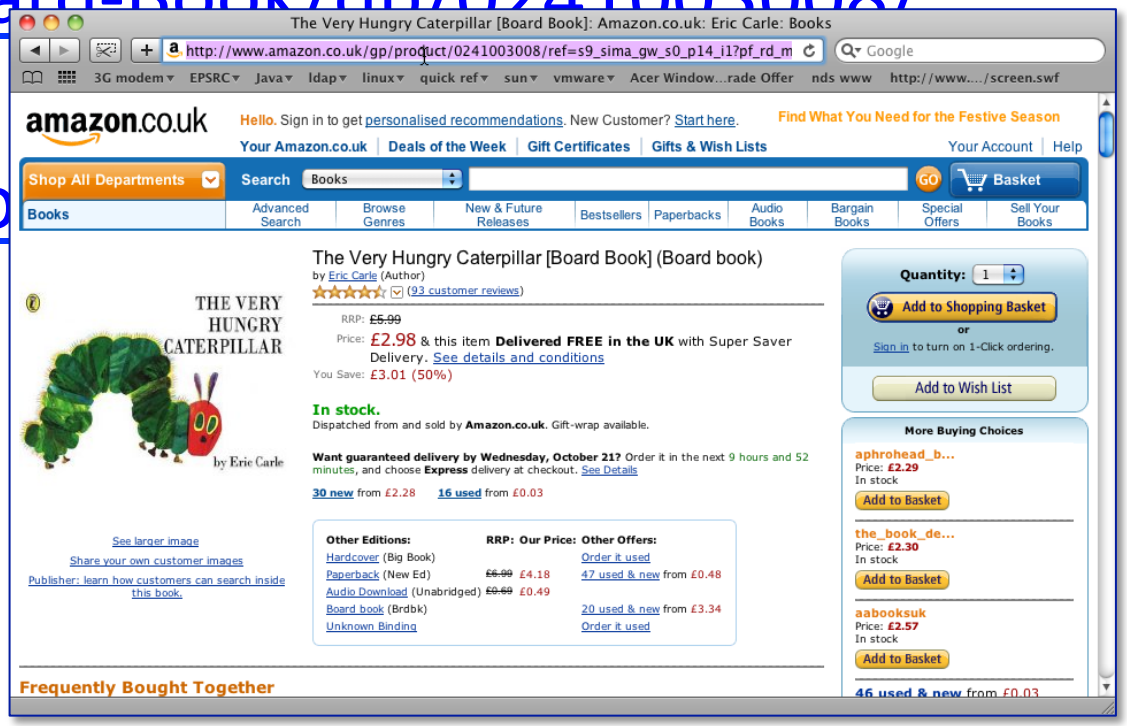
- My definition of a "name":
  A set of bits used to label an object. The semantics of the name are defined within the context of use of the object it names.
- Examples:
  - protocol name - 'http'
  - port number – '80'
  - fully qualified domain name (FQDN) - 'marston.cs.st-andrews.ac.uk')
  - IP address - '138.251.195.61'

URI [1]

- Human-friendly (?) names for resources:
  http://www.cs.st-andrews.ac.uk/
  http://138.251.206.45/
- Conventions
  https://mars
- Protocol usa
  http://www.

URI [2]

- URIs can get very complex: http://www.amazon.co.uk/Very-Hungry-Caterpillar-Board-Book/dp/0241003008/ref=sr_1_1?ie=UTF8&s=bo



Middleware - CORBA

- CORBA allows naming of IORs:
  corbaloc:iiop:1.2@srv1.blob.org:3075/NameService
- CORBA allows naming of name services:
  corbaname:blob.glob.org:2809/NameService#x/y/z
- IP addresses can also be used:
  corbaname:10.12.14.16:2809/NameService#x/y/z
- Notice the use of a DNS name or IP address.

User programs - Java API

- TCP Client:
  Socket skt = new Socket("srv.blob.com", 1234);
- Can also use an IP address:
  Socket skt = new Socket("10.12.14.16", 1234);
- Notice, once again, the use of either a DNS name or an IP address.

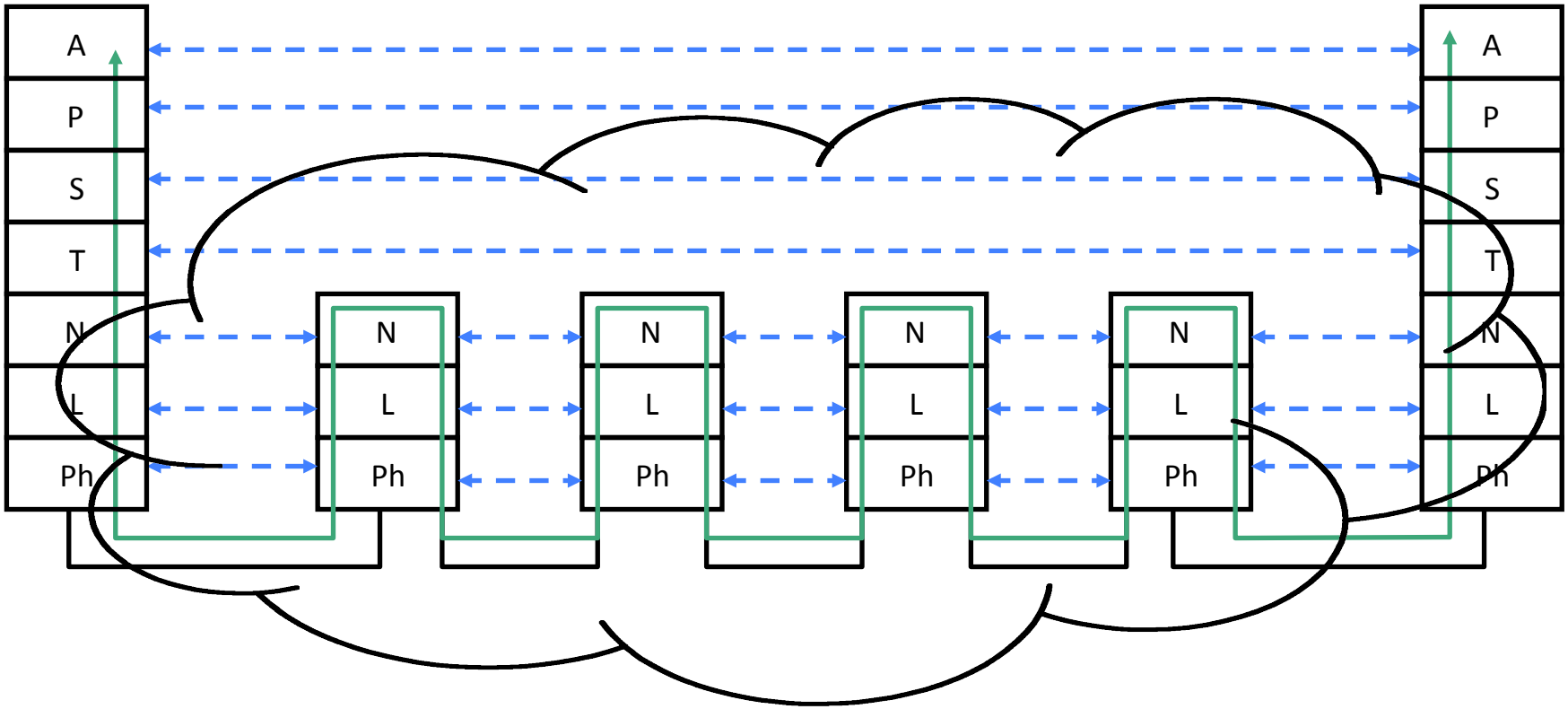
Layering – names for networking

- Functions required for object names:
  - identity
  - location ("routing", e.g. email, p2p)
  - session control
  - service differentiation

Layer	Functions
Application	Service, Session, "Routing" (overlay), Identity
Transport	Identity
Network	"Routing" (global), Location
Phy/MAC	Identity

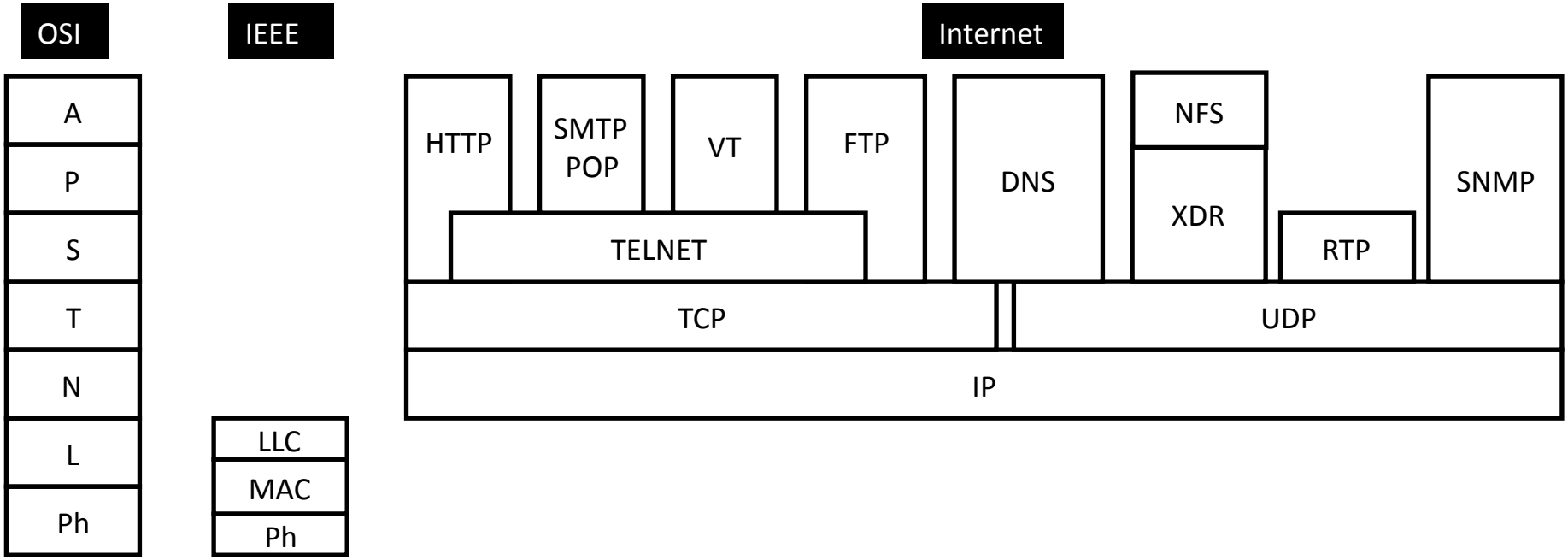
Layering

Layers for architecture



A	application	N	network	logical (peer-to-peer) communication
P	presentation	L	(data) link	actual end-to-end message path
S	session	Ph	physical	
T	transport			PRM protocol reference model

Models vs. Reality

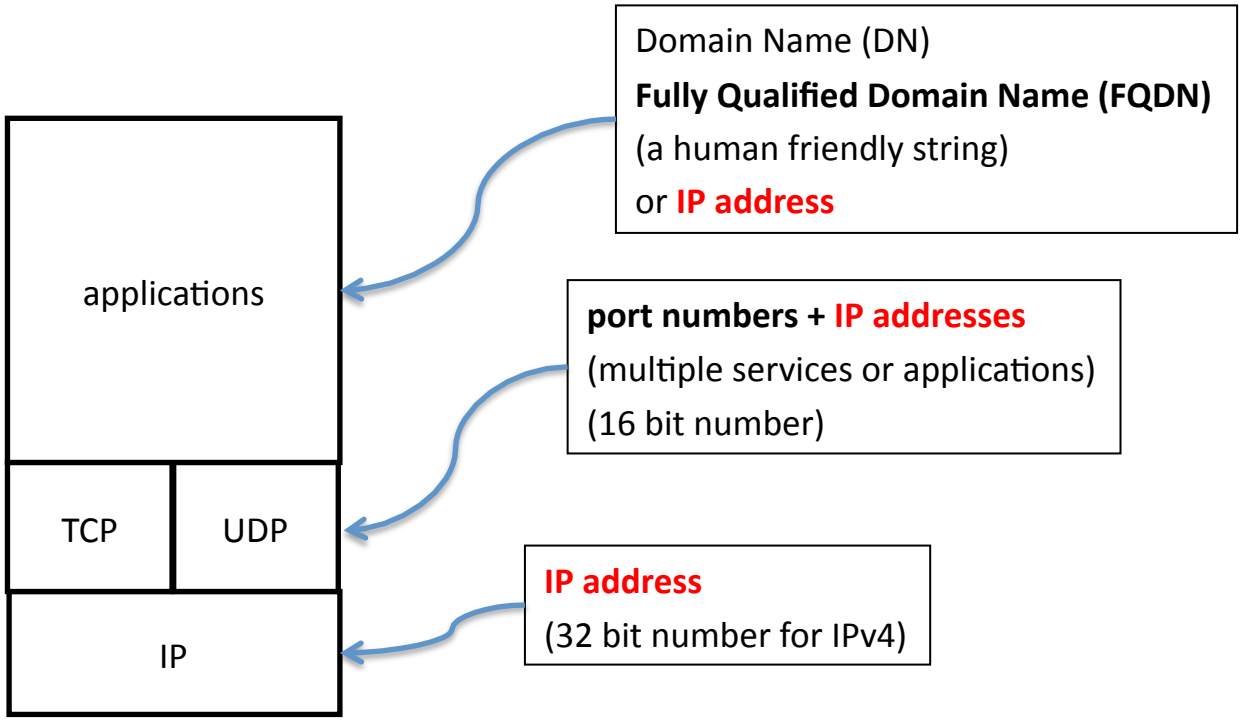


DNS	Domain Name Service
HTTP	HyperText Transfer Protocol
FTP	File Transfer Protocol
IP	Internet Protocol
IEEE	Institute of Electrical and Electronic Engineers
LLC	Logical Link Control
MAC	Medium Access Control
NFS	Network File System
OSI	Open Systems Interconnection
POP	Post Office Protocol
RTP	Real-time Transport Protocol
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VT	Virtual Terminal
XDR	eXternal Data Representation

Layering

- Split complexity of large problem across layers
- Split functionality across layers:
  - bit transmission and framing (phy/MAC)
  - global routing (network)
  - end-to-end communication (transport)
  - session control and service control (application)
- Layers can operate independently:
  - well-defined interfaces between layers
- **End-to-end capability**

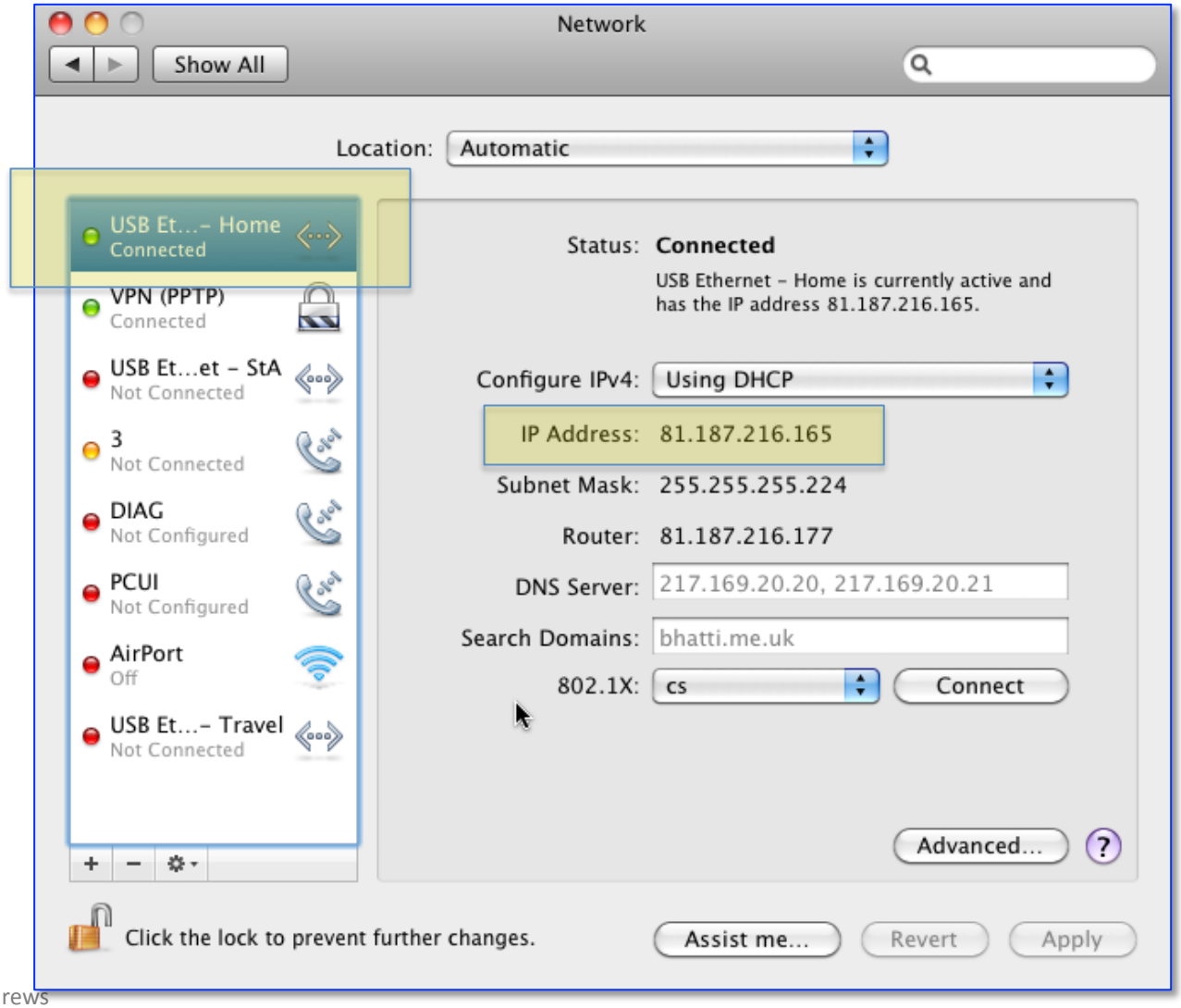
Name/Object bindings



A FQDN is looked up in a global directory service (the **Domain Name System**) and a corresponding IP address is found.

Strictly speaking, an IP address identifies an **interface on a host** (such as an Ethernet interface or wireless LAN interface) and **not** the host itself.

Interface names

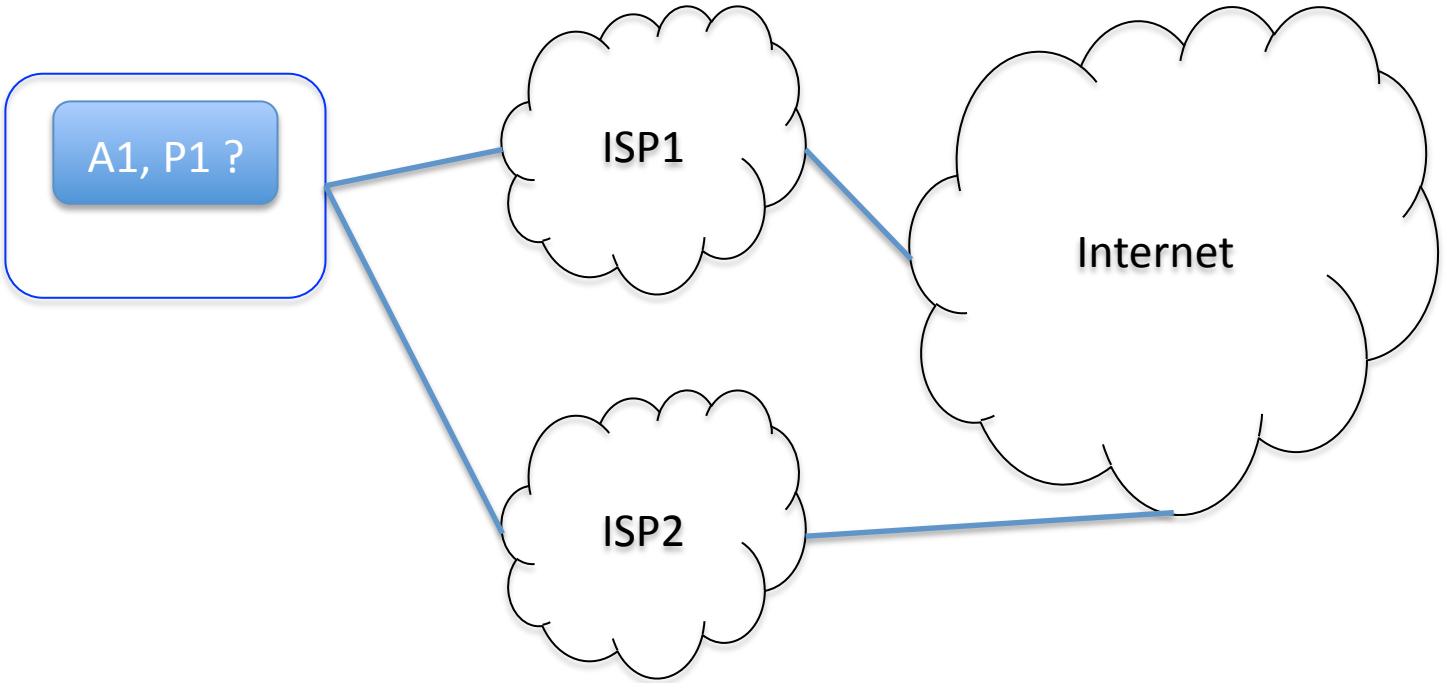


Layers entangled

Layer	IP
Application	IP address or FQDN
Transport	IP address + port no.
Network	IP address
(Interface)	IP address

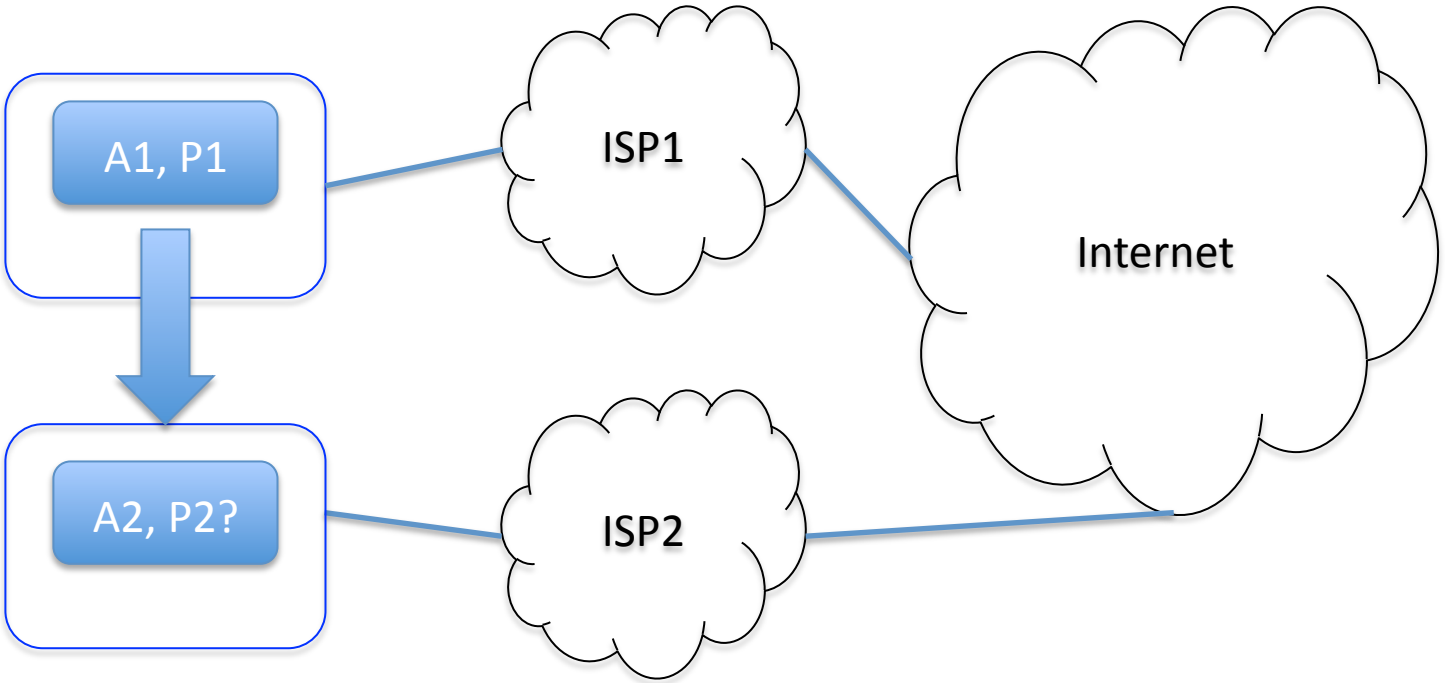
This is a serious problem …

Example: VM multi-homing



A1 belongs to ISP1 or ISP2 or neither?
How does the TCP connection identify itself?
(Today, handled through some hackery and proxies …)

Example: VM migration (mobility)



How does the TCP connection (re-)identify itself at the new location?
(Today, handled through some hackery and proxies …)

Problems due to naming today

- Multi-homing
- Mobility
- Localised addressing:
  - Network Address (Port) Translation (NAT/NAPT)
  - Network 'realms' (administrative boundaries)
- Packet-level, end-to-end security
- …

The IP address is “wrong” [1]

RFC1958 (1996) Architectural Principles of the Internet
http://www.ietf.org/rfc/rfc1958.txt

The key to global connectivity is the inter-networking layer. The key to exploiting this layer over diverse hardware providing global connectivity is the "end to end argument".*

Avoid any design that requires addresses to be hard coded or stored on non-volatile storage (except of course where this is an essential requirement as in a name server or configuration server). In general, user applications should use names rather than addresses.

* J. H. Saltzer, D. P. Reed, D. D. Clarke
“End-to-end arguments in system design”
ACM Transactions on Computer Systems (TOCS) archive
Volume 2 , Issue 4 (November 1984), Pages: 277 – 288, ISSN:0734-2071
http://portal.acm.org/citation.cfm?id=357402

The IP address is “wrong” [2]

RFC4984 (2007) Report from the IAB Workshop on Routing and Addressing
http://www.ietf.org/rfc/rfc4984.txt

A refinement to Rekhter's Law, then, is that for the Internet routing system to scale, an IP address must be assigned in such a way that it is congruent with the Internet's topology. However, identifiers are typically assigned based upon organizational (not topological) structure and have stability as a desirable property, a "natural incongruence" arises. As a result, it is difficult (if not impossible) to make a single number space serve both purposes efficiently.

Following the logic of the previous paragraphs, workshop participants concluded that the so-called "locator/identifier overload" of the IP address semantics is one of the causes of the routing scalability problem as we see today. Thus, a "split" seems necessary to scale the routing system, although how to actually architect and implement such a split was not explored in detail.

Caching

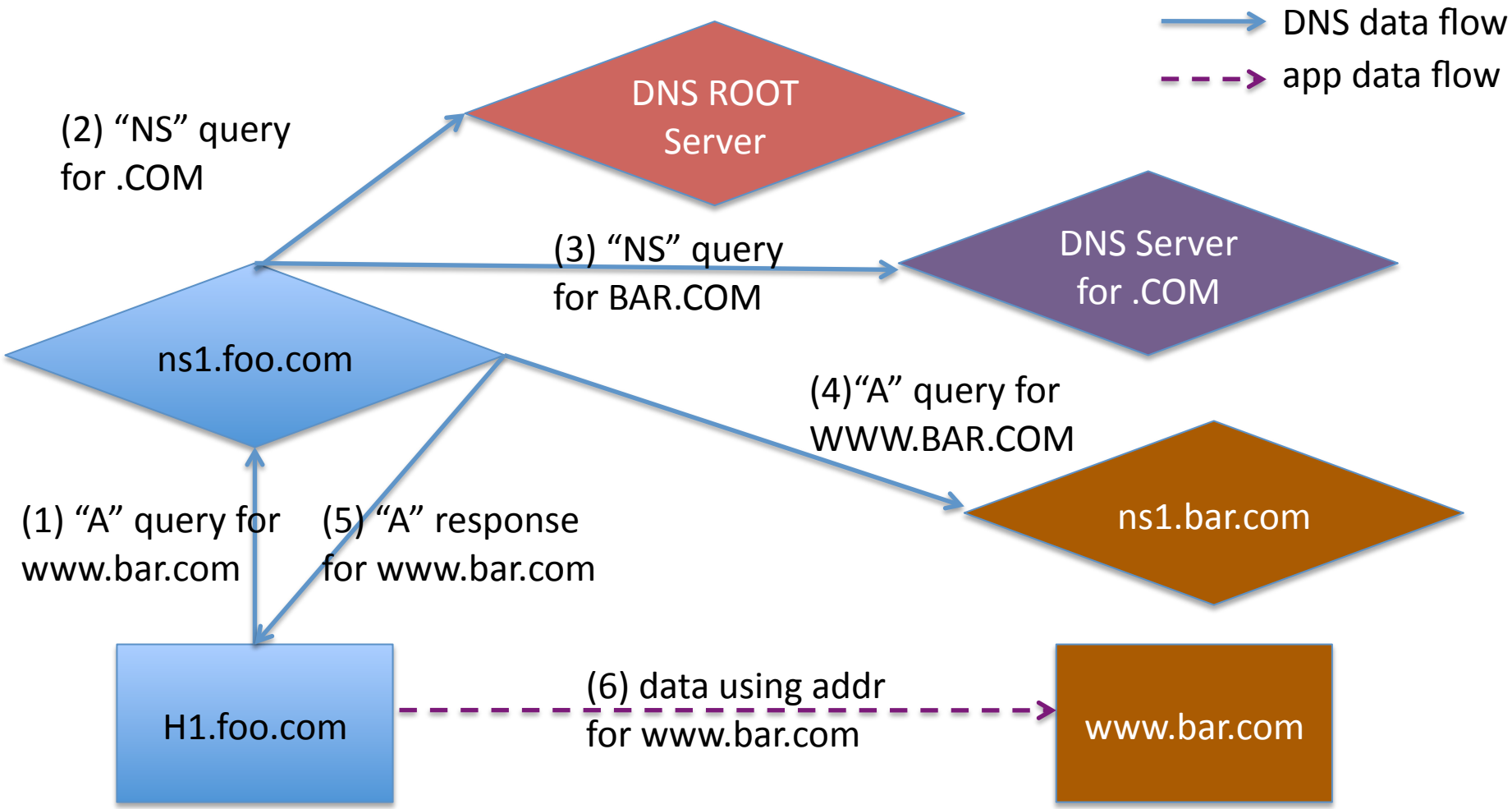
Domain Name Service

- 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 Q1/2008
- Modify TTL values of records in operational DNS server at School of CS, St Andrews
  - 2 DNS servers: BIND 8.2.4 running on Linux
  - ~400 DNS clients: BSD, Linux, Mac, & Windows
- TTL values for successive 7-day periods during normal semester:
  - Changed DNS TTL on BIND
  - used TTL values 1800s, 60s, 30s

DNS experiments at StA [2]

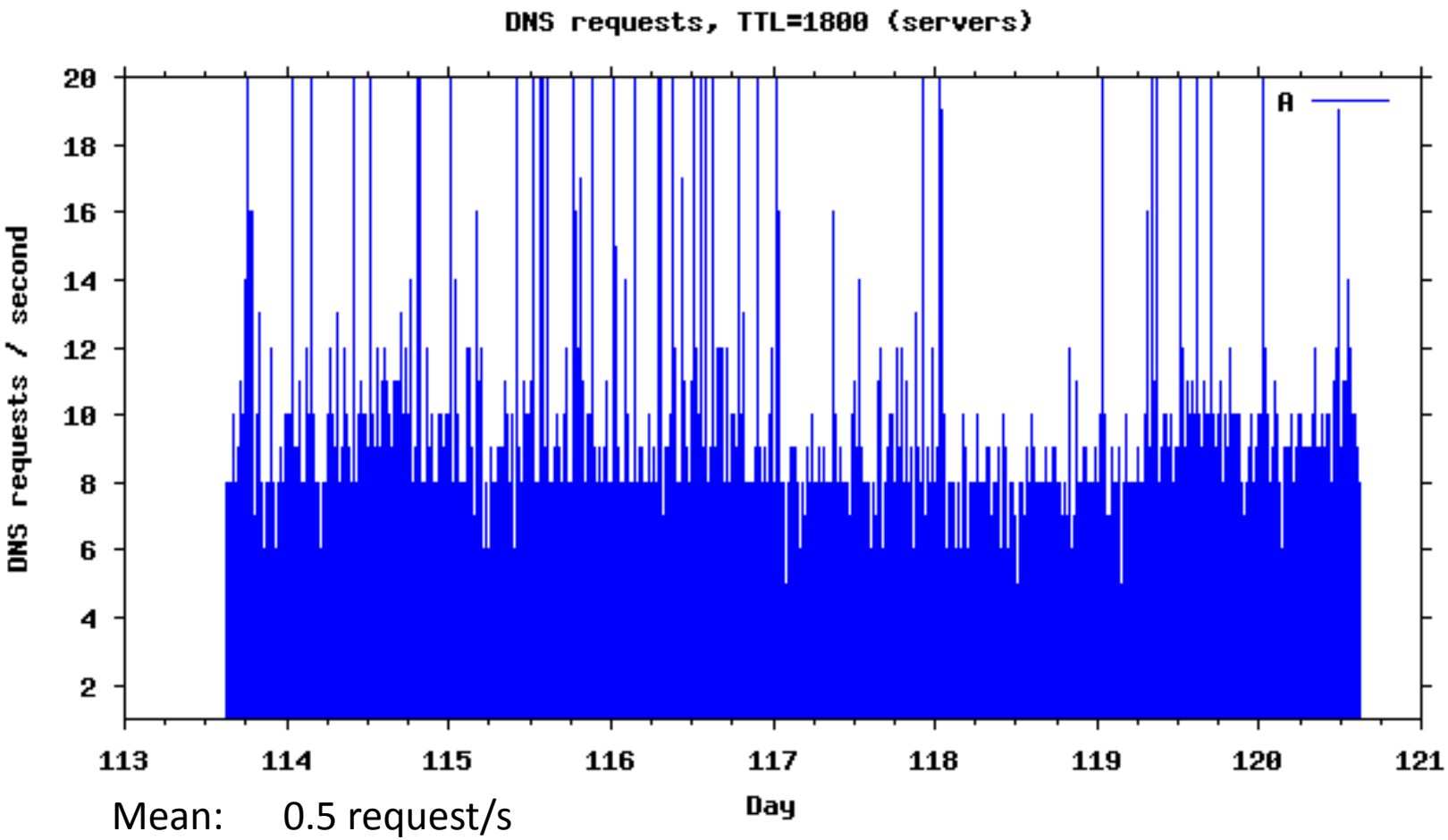
- Passive collection of packets via port mirror:
  - *tcpdump(8)* targeting *port 53*
  - Captured all DNS packets
- Results shown on following slides are for:
  - A record requests for servers only during the capture period (relevant to ILNP, and less 'noisy' data)
  - 1 second buckets
- Basic statistics: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

Basic dataset meta-data

Data set name	TTL [s]	Duration [s][1]	Total DNS packets captured[2]	Number of A record requests for 44 servers[3]
dns1800	1800	604,740	9,841,469	303,442
dns60	60	604,739	10,420,760	609,811
dns30	30	604,800	10,979,131	911,537

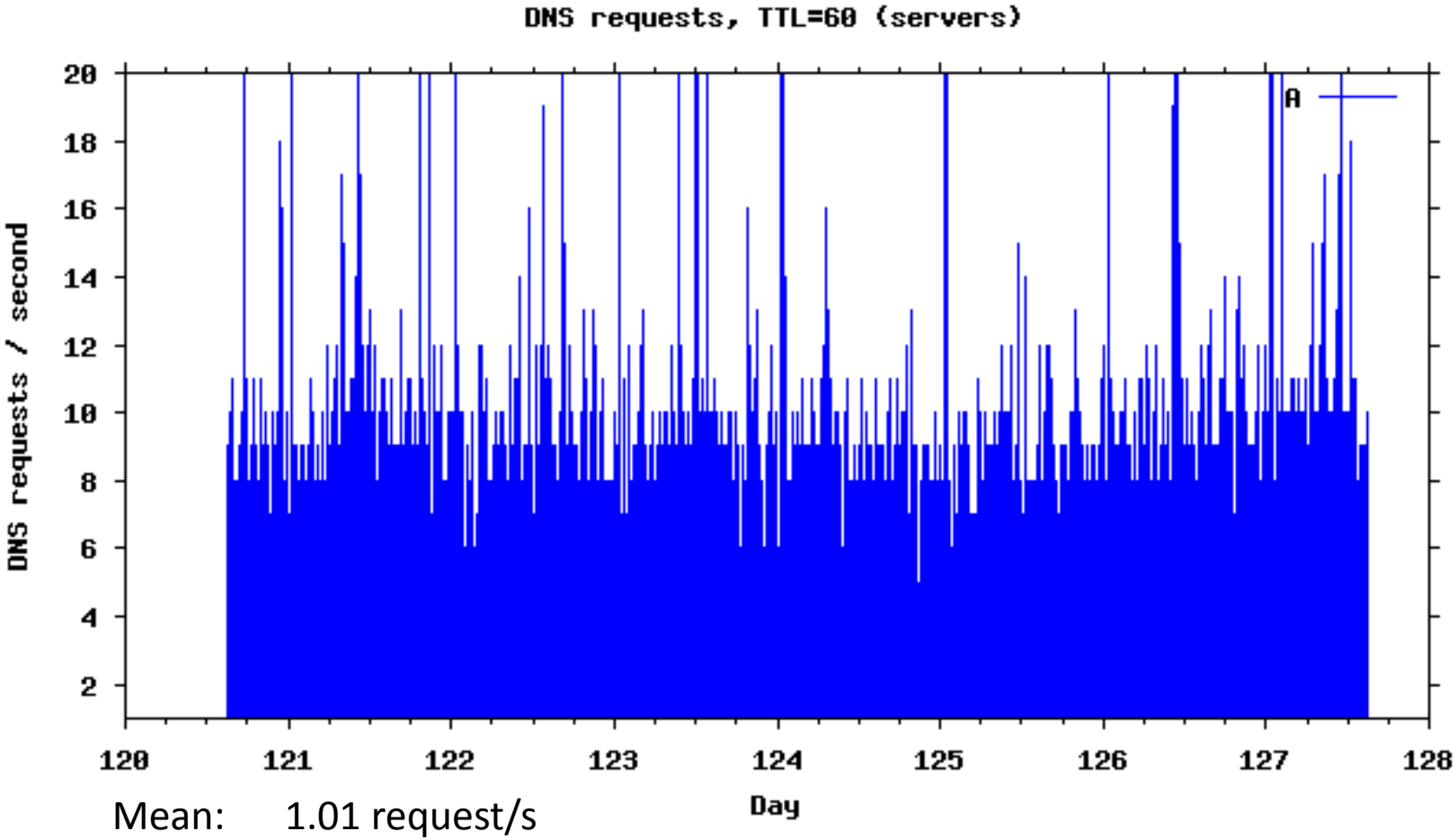
[1] from tcpdump timestamps, rounded to nearest second, 7 days = 604,800 seconds
[2] includes all request and response packets to/from port 53 (TCP and UDP), including erroneous requests etc
[3] servers that were active during the 3 weeks of data capture

dns1800: A record requests TTL=1800s



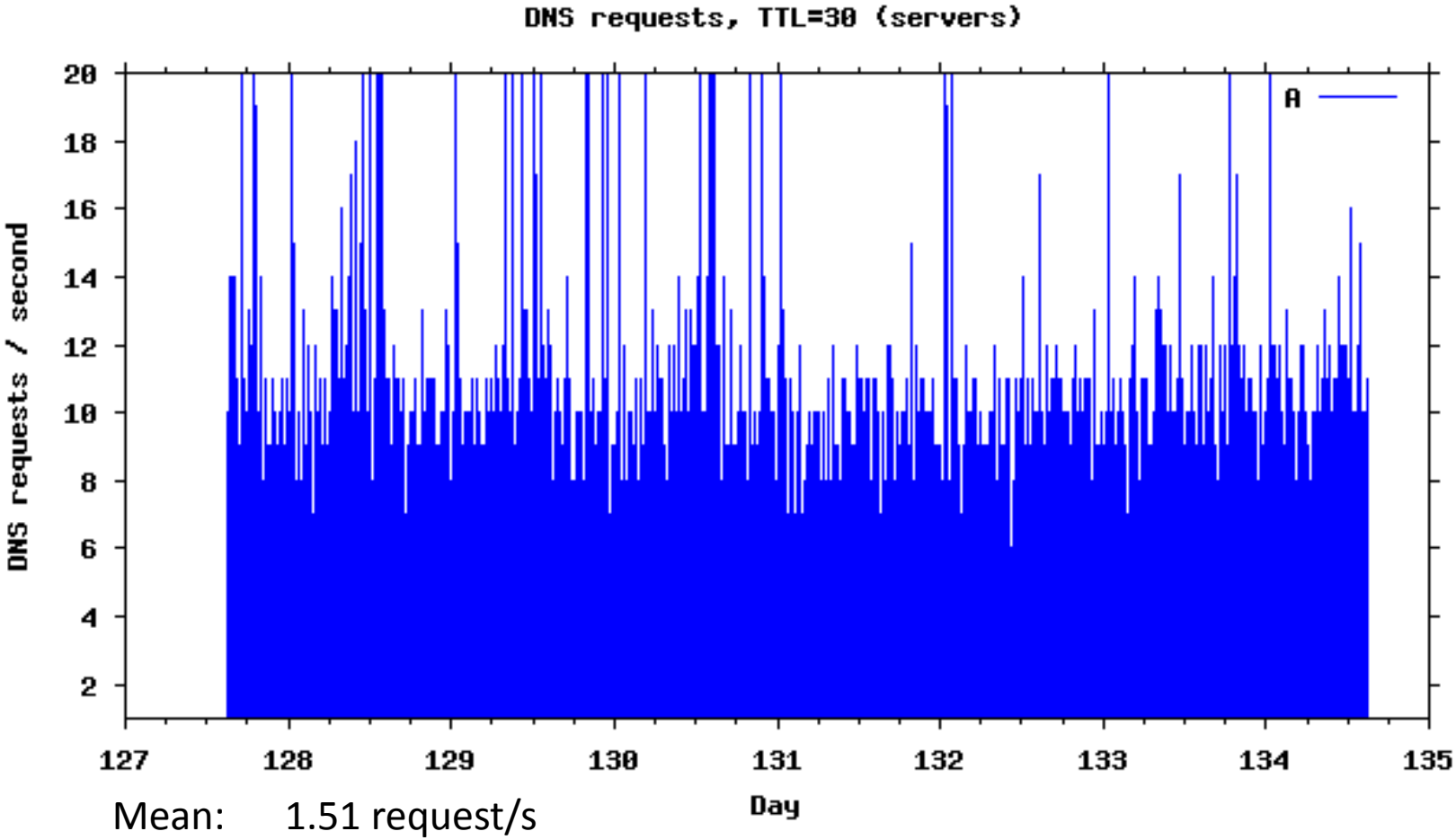
Mean: 0.5 request/s
Std Dev: 1.17 requests/s
Max: 82 requests/s

dns60: A record requests TTL=60s



Mean: 1.01 request/s
Std Dev: 1.29 requests/s
Max: 67 requests/s

dns30: A record requests TTL=30s



Mean: 1.51 request/s
Std Dev: 1.54 requests/s
Max: 116 requests/s

Summary of basic statistics

Data set name	Mean [request/s]	Std Dev [requests/s]	Maximum [requests/s]
dns1800	0.50	1.17	82
dns60	1.01	1.29	67
dns30	1.51	1.54	116

60 fold drop in TTL values
results in (only)
3 fold increase in A record requests

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
- High-speed load balancing and VM mgmt for data centres
- Support for mobility and multi-homing:
  - Location updates give changes in connectivity
- Help defend against certain network attacks:
  - DNS cache poisoning for end-sites
  - DDoS: fast-cycle multi-homing (i.e. a kind of "fast-flux" DNS for defence rather than attack)
  - Others possible …
- Potential for edge-site based multi-path and TE control:
  - multiple Locator values and DNS L record preferences

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 (mobile) site can remote some or all of its authoritative DNS servers; some sites do so today.

Ongoing DNS experiments at StA

- More measurements in Q1/2009:
  - Site is now using Microsoft Active Directory for its DNS servers
  - Tuning of client-side OS and browser caches to reduce end-system caching effects on DNS data
- Q1/2009 experiments:
  - using TTL values of 1800s, 60s, 30s, 15s, 0s
  - ~250GB of data collected so far
  - analysis in progress …

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

Summary and Conclusion

- Summary:
  - **Naming** has become 'untidy', lacking clear semantics
  - **Layering** has become entangled due to the use of the IP address in many different layers
  - **Caching** is not as effective as people think it should be
- Conclusion:
  - We need to re-think our approach to the use of names across the layers: ***make the naming cleaner***
  - Exploit new functionality that is possible through a cleaner approach to naming