USING ~300 BILLION DNS QUERIES TO ANALYSE THE NAME COLLISION PROBLEM

Jim Reid, RTFM LLP
jim@rfc1035.com

Background

- ICANN concerned about potential problems from new gTLDs clashing with existing *ad-hoc* use of these in domain names, "private" name spaces and certificates
  - Some anecdotal evidence, but no hard data
- Study approved by ICANN board in mid May 2013
  - Is there a problem?
  - If so, how big is it?
  - What risk mitigation frameworks could be applied?

Timing

- **VERY** Ambitious!
- Find, gather & analyse data
  - First find out how best to do that and what resources can be brought to bear
- Report by Durban ICANN meeting ~6 weeks away
  - Expect findings to be challenged/attacked/checked
  - Light the touchpaper and watch the firework display…
- Got even scarier once the scope of the data crunching became apparent

Objectives for DNS Component of the Study

- Count how often new gTLDs appear in root server traffic
  - Are these requests localised or diffuse?
  - Proper resolving servers or from forwarders/stubs?
  - How does this compare to traffic for existing TLDs?
- How often do new gTLD labels appear elsewhere in QNAMEs?
  - Where do they appear?
- For bonus points, look at big resolver operators' traffic

Kick-Off

- Preliminary discussions took place at RIPE66 in Dublin
  - Many RSOs present, DNS-OARC meeting too
- Solution: use the yearly DITL (Day in the Life of the Internet) datasets at DNS-OARC
  - ~1 day of root server DNS traffic as pcap files
- Only practical way to get access to suitable data
  - Simple, quick fix for privacy and data protection concerns

Initial Scoping

- Helpful advice and software from Netnod
- Got access to elderly box, **`an1.dns-oarc.net`**
  - 2-core 1Ghz Opteron, 2GB RAM, limited local disk
- Did some prototyping with **`packetq`**
- Some nasty shocks:
  - ~1000 new gTLDs found in a sample of the DITL pcaps
  - 1 pass over the 6TB of DITL pcaps for 2012 would take at least 2 weeks on this system: far too long

CAIDA to the Rescue

- Lot of uncertainty over what other hardware could be provided:
  - Could anything be ordered, delivered and set up in time?
  - Maybe NFS mount the datasets into the cloud somewhere?
    - Throw a bazillion CPUs at the problem
- Found out CAIDA had a server which could be made available
  - 8-core 2GHz Xeon, 7TB of scratch disk space
  - Running 5-6yo version of FreeBSD
  - I pass over a year's DITL data would take less than a week

Software Choices

- Got a custom version of **`packetq`** from Netnod
  - SQL-like language for crunching through pcap files
  - Mostly counted things: QTYPEs, QNAMEs, source addresses
  - Not so good for label position counting/checking though
    - 1 week of CPU time for each N-th level label to inspect
- **`tcpdump`**, **`awk`** & **`fgrep`** for a second pass over pcap files
  - Second data run took 1 week of elapsed time

Software Choices - 2

- Use **`tcpdump`** & **`fgrep`** for a second pass over the pcaps
  - Generated text files containing pretty-printed DNS requests where any label matched a proposed gTLD
    - "Only" several GB of text files to then analyse
  - **`awk`**-based scripts chugged through these text files to do label position and source address prefix counts
    - Sometimes tripped over bad input data because of malformed (-ish) queries, e.g. ***`foo.bar.tld .`***

General Approach

- Split the ~250,000 pcap files for each year into 8 equal chunks
- Run script over each pcap as an "atomic" operation
  - Generate unique output files for each input file
    - Merge or aggregate these interim files later
    - Could process files by hand if bugs/corner cases pop up
  - No locking/synchronisation issues
  - Just keep crunching, never stop or go back
  - Flag errors as corner cases, but don't allow these to get in the way or complicate the scripting

Triple-Distilled Data

- 1: reduce terabytes of raw data to *O*(gigabytes) of rough results
- 2: distill rough results to *O*(megabytes) of refined results
- 3: feed refined results into spreadsheets and PHP-based tools for statistical analysis
  - Summary results analysed in more detail by Interisle
    - Some sampling done too
  - Interisle drew graphs and compiled tables for final report

A Counting Problem

- DITL data sets had hundreds of millions of 10-character strings as TLD queries, each string used only once
  - What was this, DNS queries as a covert channel?
- Needed to count **ALL** of these, just in case one string turned out to be heavily used
- Turned out to be a Chrome feature
  - Looks up non-existent TLD to see if local resolver server does NXDOMAIN rewriting or similar Stupid DNS tricks™

Why no **`perl`** or **`python`** or…?

- CAIDA box had old versions of these
  - Incompatible with latest **`perl/python/`*`whatever`*** tools
- GNU autoconf nested dependency hell
  - Couldn’t blooter existing stuff in case that affected the CAIDA users who’d lent out the box
- Had to ask for latest **`g++`** compiler for **`packetq`**
  - Couldn’t impose on sysadmin for even more goodwill

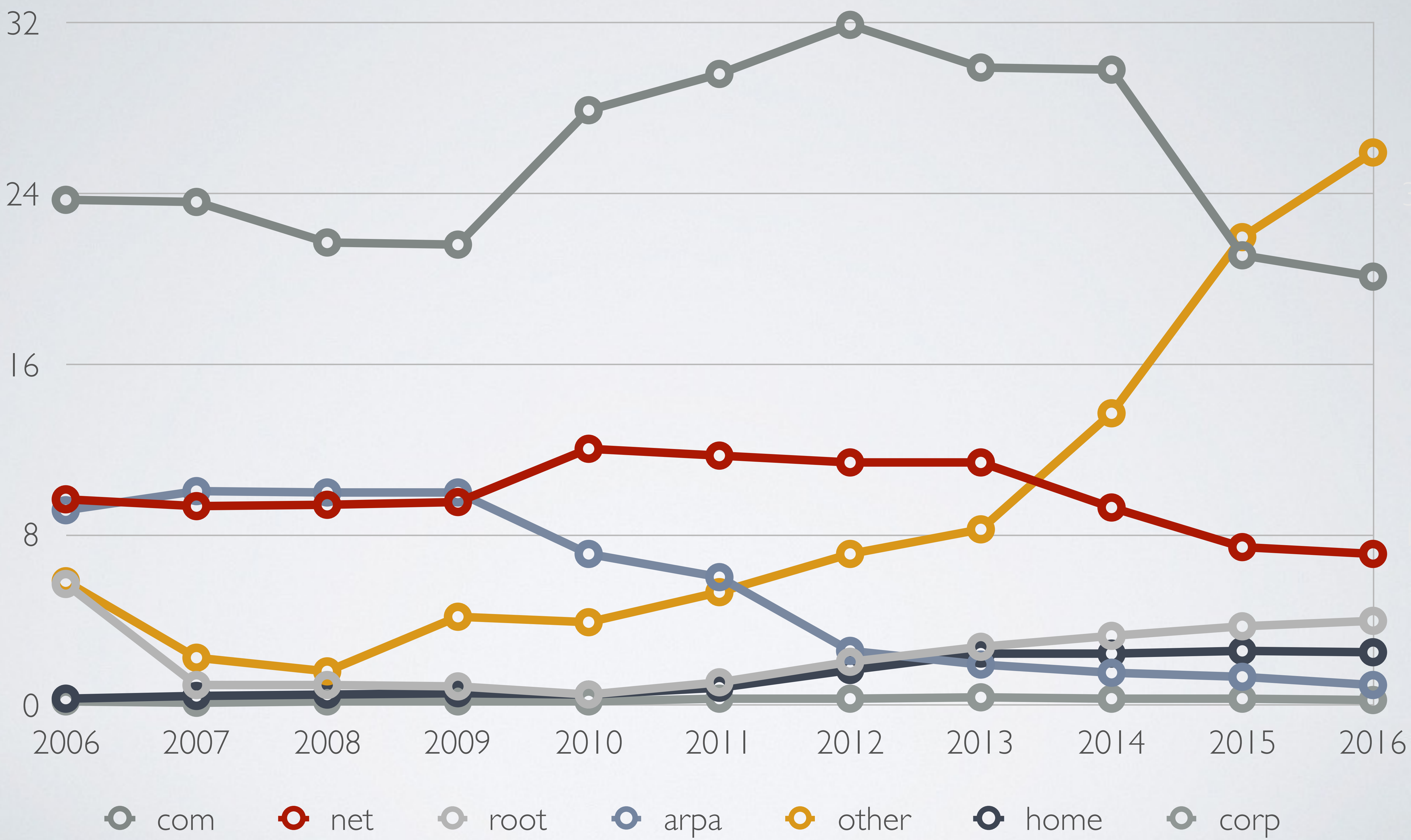
Why no Database?

- Couldn't realistically prototype/calibrate this in time
- Far too many unknowns
  - How big would the database(s) be?
    - What's the optimal size of the tables and indexes?
  - How long would it take to populate the database(s)?
    - Locking/synchronisation issues with 8 CPUs in parallel
  - How long would SQL queries take to run?
  - What if the database got corrupted?

Findings

- Lots of power-law distributions
  - Small numbers of TLDs and source addresses (per TLD) accounted for most of the traffic
- **FAR** more traffic for proposed TLDs than gut feel suggested
  - Almost all new gTLDs were seen
  - Traffic for **`.home`** and **`.corp`** was particularly high
- Pretty much none of that DNS traffic was localised (enough)
- Some interesting/unexplained traffic patterns

TLD Queries as %age DITL Traffic



For Further Analysis?

- Probable leakage from Active Directory and Bonjour
  - How will those end systems behave if/when NXDOMAIN becomes a referral response?
- Some dynamic updates too....
- Lookups for MX and SRV records
  - Can't be coming from naive end users & applications
  - Something's been deliberately (mis)configured to look for these: what? why?
- These deserved follow-up study, but didn't get it

The “Safe” Query Rate Threshold

- Lot of undue comment and attention on this
  - ICANN’s choice as the only metric
- The **`.bv`** and **`.sj`** ccTLDs were empty and unused in 2013
  - Nobody has a valid operational reason for querying them
  - Traffic volume they get seems a fair indication of the DNS background noise level as seen in root server traffic
- This was only one metric out of many and might well not be the most significant one for assessing new gTLD “safety”

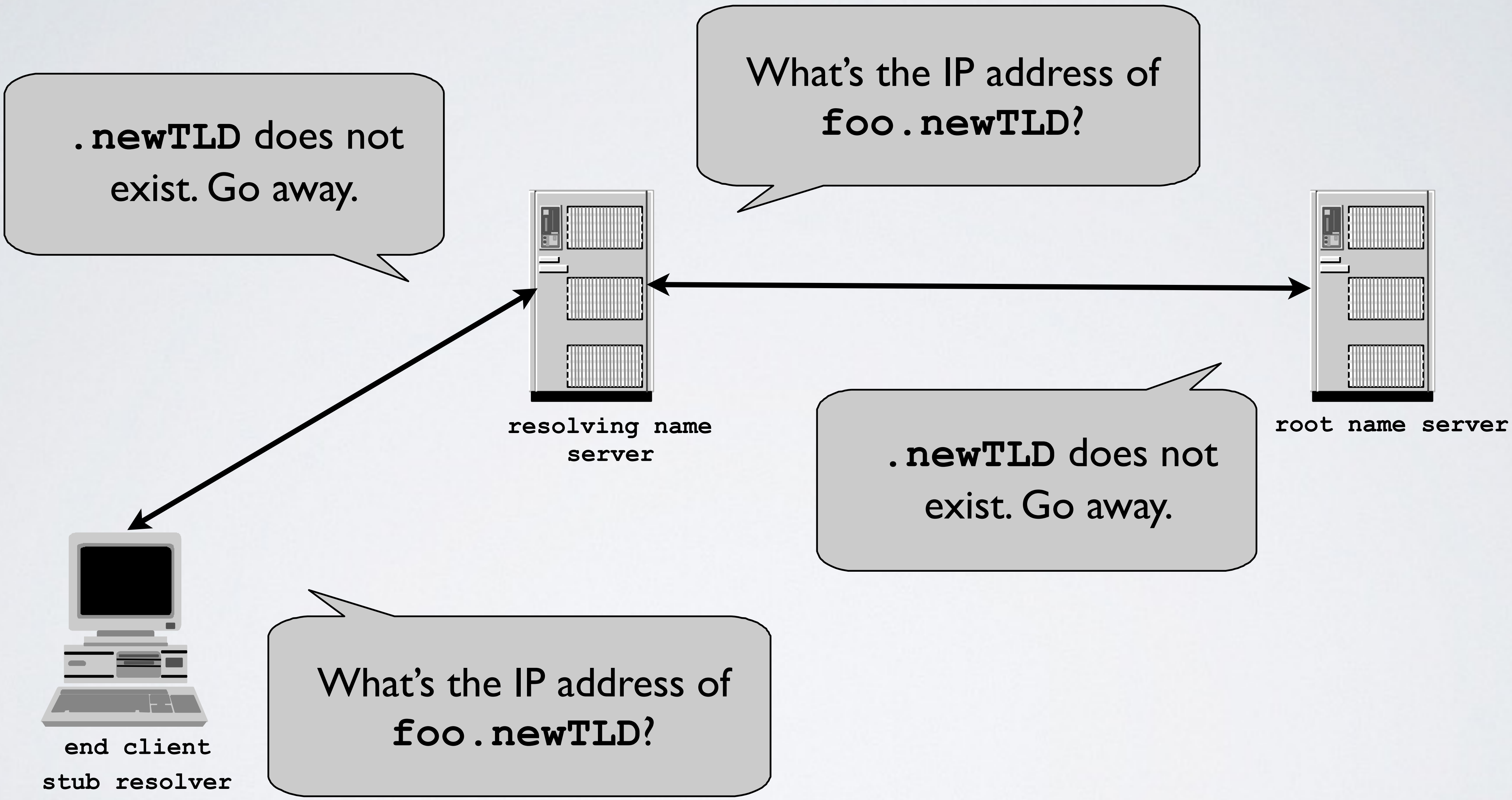
ICANN Risk Mitigation Strategy

- **`.home`** and **`.corp`** are effectively dead
  - **`.mail`** added to ICANN's dead pile too
- Other gTLDs can proceed to delegation
  - Wildcard everything for 90 days:
- **`*.gTLD. IN A 127.0.53.53`**
- **`*.gTLD. IN TXT "Your DNS is broken..."`**

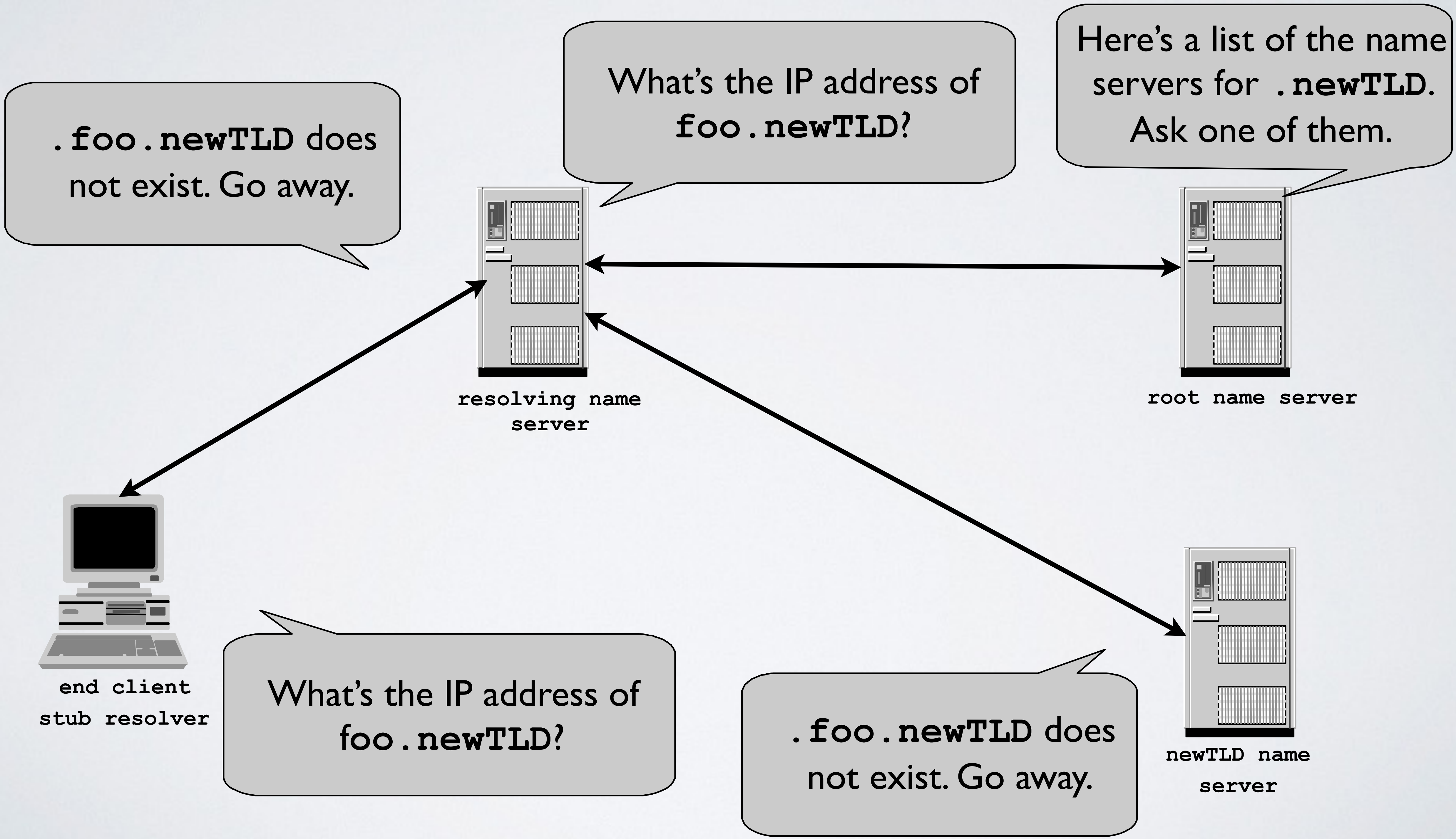
Mitigating Name Collision: Early ICANN Approach

- If ***whatever.newTLD*** appears in DITL data, just arrange for the ***.newTLD*** name servers to return NXDOMAIN
  - Lookups for ***whatever.newTLD*** continue to get NXDOMAIN responses, just like now
- DNS behaviour is unchanged, so no problem… maybe
  - It used to be the root servers that return NXDOMAIN, but once ***.newTLD*** is delegated, its name servers do that
- Is this strategy prudent or not?

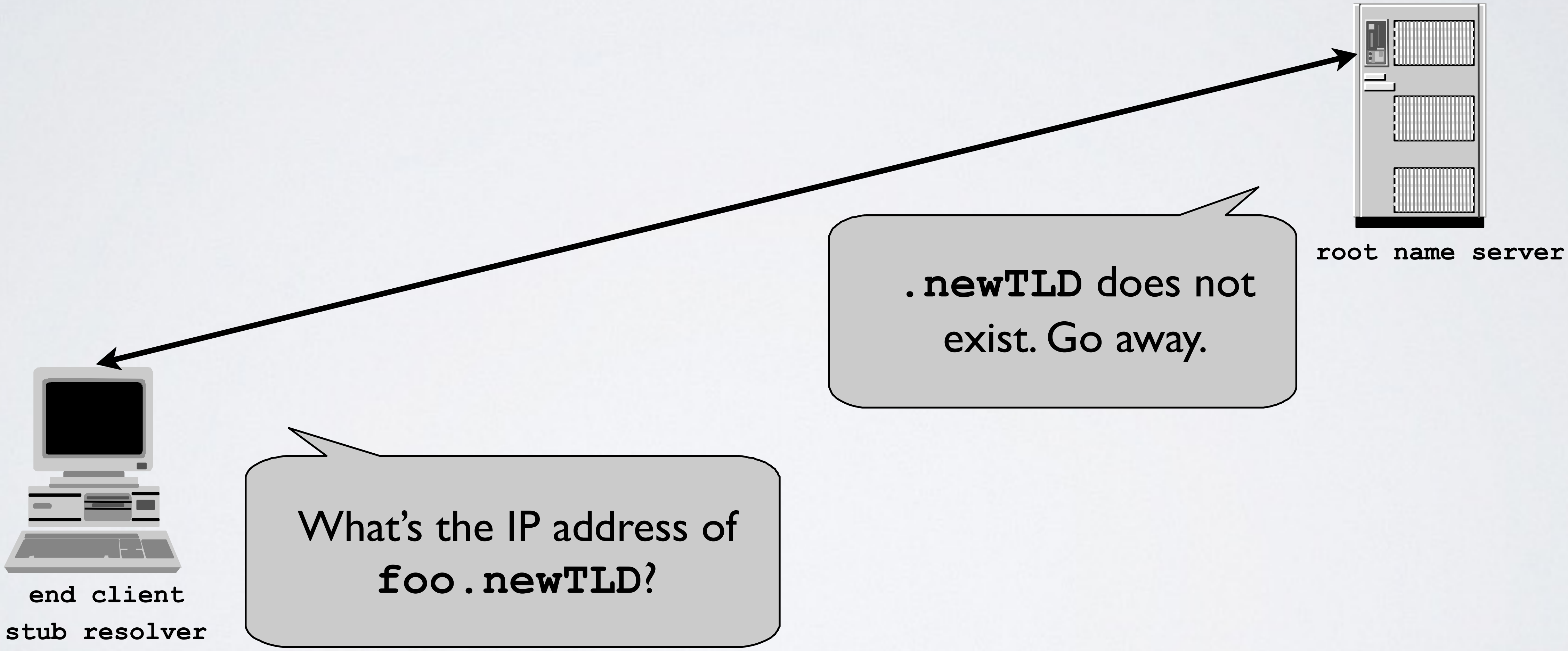
A conventional DNS lookup before `.newTLD` is delegated



A conventional DNS lookup after .newTLD is delegated



An unconventional DNS lookup before `.newTLD` is delegated



An unconventional DNS lookup after `.newTLD` is delegated

Naive DNS Clients

- Stub resolvers, proxies & forwarding-only servers cannot handle referral responses
- Undefined behaviour when they get referrals:
  - Give up, report an error, try another name, fail, crash....
- These devices sometimes mistakenly query the root
  - How often does this happen?
  - Is it a problem or not?
  - Which TLDs are most/least at risk?

Analysis & Crunching

- Chewed through ~10TB of DITL data: ~200Bn requests
  - Contributing root server pcaps from 2006-2013
  - Made three passes over that data
- Qualitative analysis
- Comparitive analysis
- Historical analysis
- Qualitative analysis

Quantitative Analysis

- There's quite a lot of RD=1 request traffic already
  - Around 12% ± 5% of current root server requests
  - Not supposed to happen
    - Only resolving name servers should be querying the root
  - Does this appear to be causing any operational problems?
- Almost nothing does RA=1
  - No surprise: only **answering** servers are expected to set this header bit

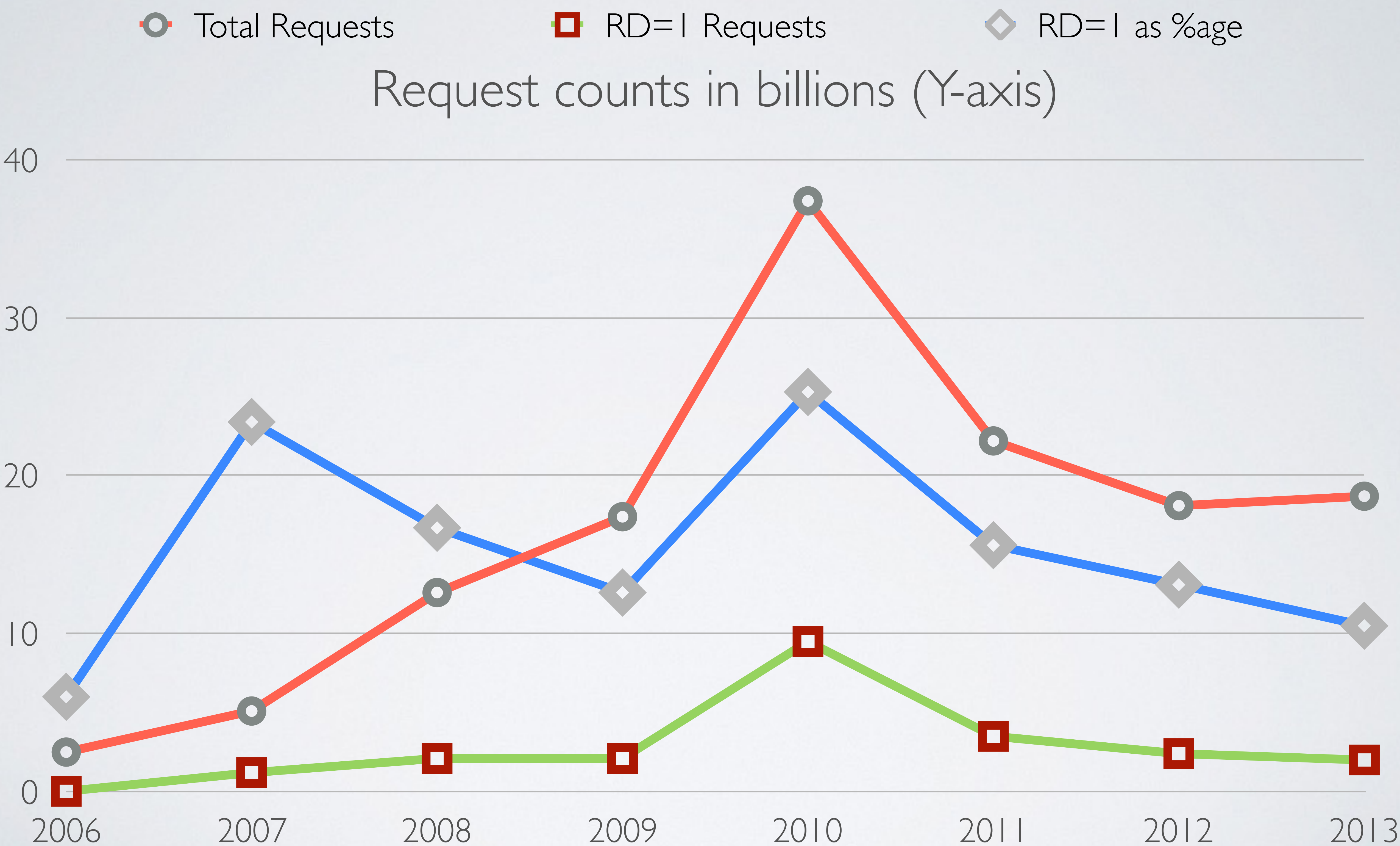
Comparitive Analysis

- Usual suspects amongst existing TLDs responsible for the majority of RD=1 requests:
  - **`.com`**, **`.net`**, **`.arpa`**, **`.org`**, **`.uk`**, **`.de`**, **`.cn`**, **`.jp`**
- Very few new gTLDs have RD=1 requests
  - **`.home`** and **`.corp`** are by far the biggest source
  - Most have none
  - Rates for the new gTLDS are generally 4 or more orders of magnitude fewer than for existing TLDs
  - **`.google`** seems to get more than its fair share

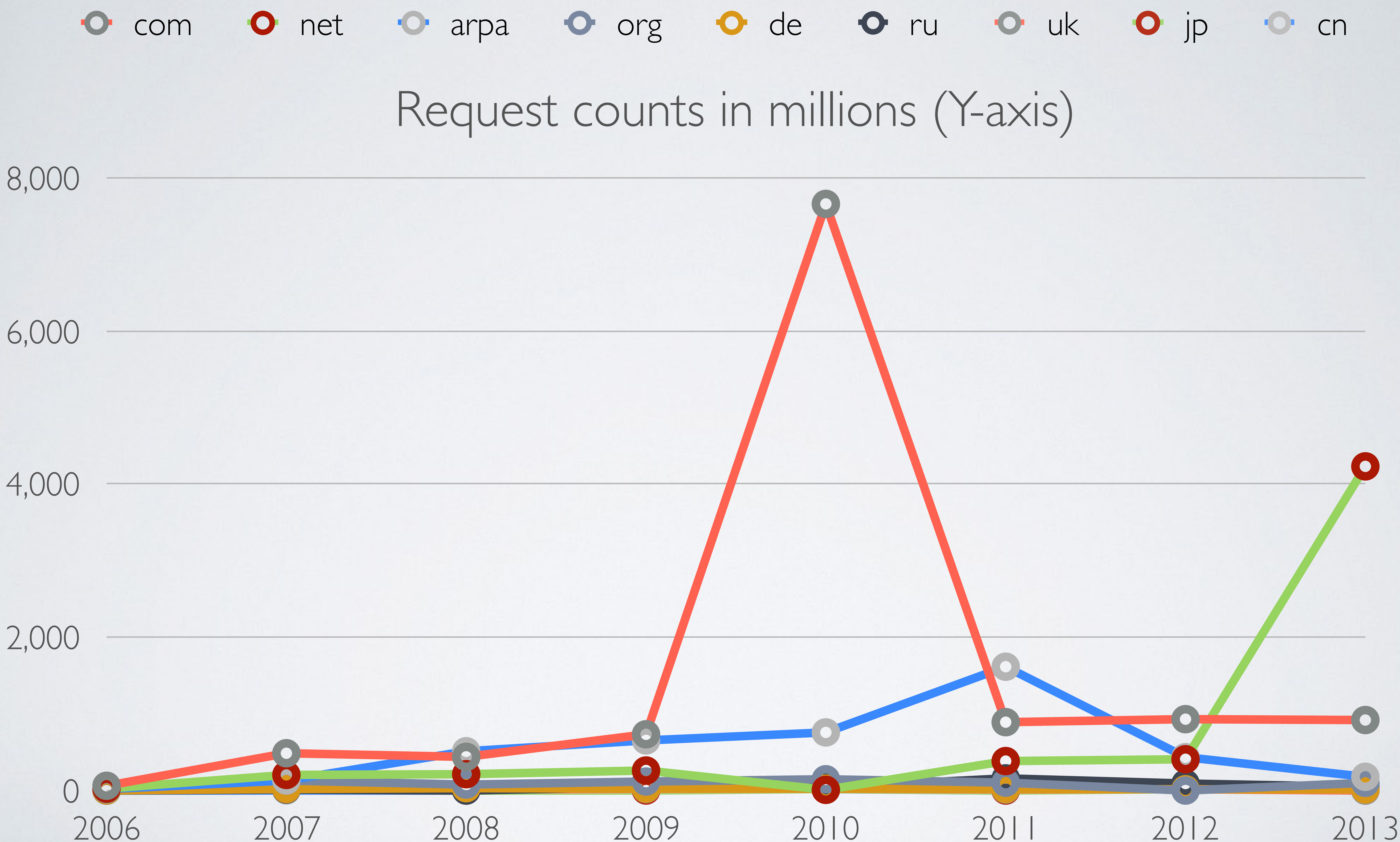
Historical Analysis

- Overall traffic patterns seem stable
- Little variation in each year's DITL data
  - Same TLDs appear in broadly the same position each year
- Behaviour of the DNS as a whole seems consistent
  - A few outliers
- Not much sign of "new/changed stuff" perturbing the observed RD=1 traffic in the DITL data sets

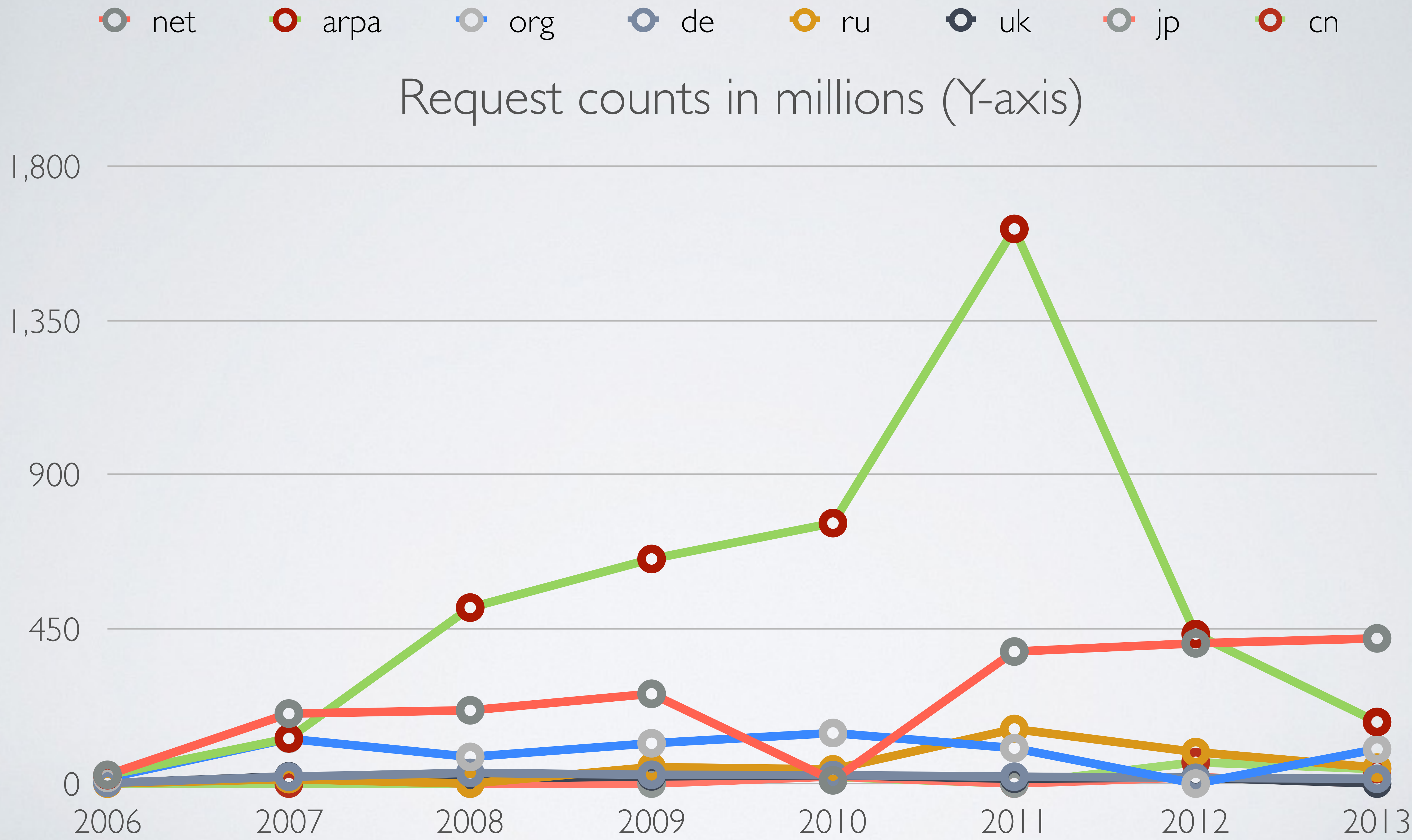
Overall RD=1 Rates/Percentages



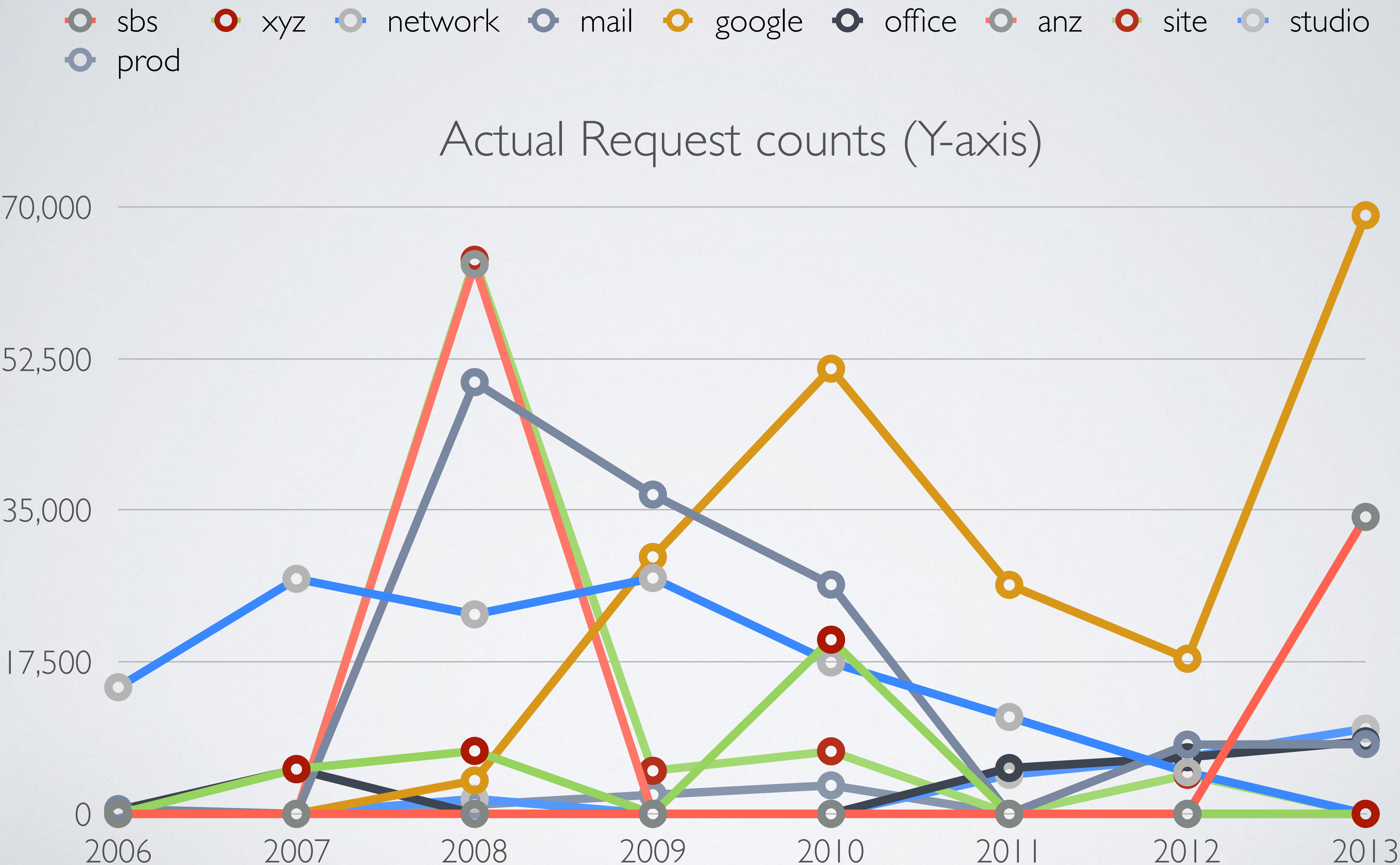
RD=1 Rates for Current TLDS



RD=1 Rates excluding .com



RD=1 Rates for New gTLDS



Qualitative Analysis

- In-depth analysis of everything would take forever and probably wouldn't unearth anything new
- Needed to make some simplifications:
  - Just looked at the glaringly obvious outliers
  - Ignored traffic levels below ICANN's "safe" threshold - except when there was something interesting to look at
- High-level summary: nothing to see here, move along

2013 Data

- 57,000 of 70,000 RD=1 queries for **`.google`** came from one IP address, a Californian school (***`something`*****`.k12.ca.us`**)
- One IP address at a US ISP generated almost all the RD=1 lookups for **`.statefarm`**
  - Remainder had RFC1918 source addresses
  - Similar patterns for **`.thd`** and **`.sbs`** traffic
- Probably looking at isolated examples of rogue applications or misconfigured CPE
  - Unable to identify root cause(s) - so far

2012 Data

- Diffuse data sources for **`.google`** lookups:
  - ~600 /24s each generating ~600 queries
  - Some RFC1918 addresses again
- Probably not worth further investigation
  - QNAMEs generally for google's mail servers without a valid TLD suffix: e.g. **`gmail-smtp-in.l.google`**
- Transient stub resolver or mail server misconfiguration?

2008 Data - 1

- Single /24 at a Florida ISP generated half the **`.anz`** RD=1 queries
  - Gloriously bizarre QNAMEs:
    - **`asad86158676.adeli.aks4you.irmr.maliblog.sina.virusgro.ups.iranmy.sharvin.lionel00.kooliver.2game2.aminpidofsh.2mb.rozmaregi.anz`**
  - Presumably nothing to do with ANZ Bank

2008 Data - 2

- ~60,000 RD=1 queries for **`klingon.site`**
- All had the same query id - 0 - and source port
- All from the same IP address
  - Prefix assigned to University of Toronto
  - No reverse DNS
- Probably a student programming exercise gone wrong
  - Mr. Spock can't code? :-)

Findings/Conclusions - 1

- There's a **lot** of RD=1 traffic going to the root already: ~12%
  - Probably always has been and always will be...
  - This doesn't seem to be breaking anything significant
  - Naive resolvers are either failing safe or working around referral responses somehow
- Billions of referrals from the root to **`.com`**, **`.net`**, **`.arpa`**, etc. do not seem to be causing problems for naive DNS clients today

Findings/Conclusions - 2

- RD=1 traffic for new gTLDs is **much** lower in absolute and relative values than the rates found for existing TLDs
  - Whatever generates these requests for new gTLDs should somehow cope OK with referral responses - probably
- Traffic for **`.google`** might be a concern if rogue clients are not isolated incidents
- Fairly stable (but low) rate of RD=1 requests for **`.mail`**
  - Could mean some mail gets delayed or bounced
- ICANN's name blocking strategy shouldn't cause harm

Acknowledgements and Thanks

- kc claffy and Daniel Anderson at CAIDA
  - Simply couldn't have done the initial work without access to their hardware
- Henrik Levkowetz at Netnod
  - For tweaking and supporting **`packetq`**
  - Also did some sanity checking of early results
- DNS-OARC, especially William Sotomayor, for logistical support

QUESTIONS?