# DITL CRUNCHING FOR GTLD NAME COLLISION STUDY

Jim Reid
RTFM LLP/Interisle LLC

# INTRODUCTION

- Why, When & How?
- Hardware choices
- Software choices & trade-offs
- Problems
- Results/Findings
- Possible next steps

# OBJECTIVES FOR DNS ELEMENT OF THE STUDY

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

# DATA SOURCE

- DITL traffic for root servers held at DNS-OARC
- Roughly 7TB of compressed pcaps in ~500,000 files
  - 5.2 TB for 2012 (~55B requests)
  - 1.7 TB for 2013 (~39B requests)
- Just reading 1 TB of data at 20MB/s takes ~14 hours
- TLD counts for 2013 DITL traffic is 3.5GB

# INITIAL SCOPING

- Helpful advice and software from Netnod
- Got access to elderly box, **`an1.dns-oarc.net`**
- 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
    - Had to make 2 passes over both 2012 and 2013 datasets

# 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 - 1

- Got a custom version of **`packetq`** from Netnod
  - SQL-like language for crunching through pcap files
  - Mostly counted things: QTYPEs, QNAMEs, source addresses
    - Special hooks to recognise existing and proposed TLDs
- Could drive all cores flat-out simultaneously
- Not so good for label position counting/checking though
  - 1 week of CPU time for each N-th level label to inspect

# 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 DATA DISTILLATION

- 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

# 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 with 8 CPUs in parallel
  - How long would SQL queries take to run?
  - What if the database got corrupted or a scratch disk died?

# WHY TCPDUMP?

- It'll be faster than **`perl`** or **`python`** or...
- Newest DNS tools need newer **`perl`**/**`python`**/***`whatever`*** versions than the CAIDA box had
  - Too many unknowns - install/software configuration hell
  - Needed certainty when results could be expected
- Needed to capture several things about requests of interest
  - QNAME, source address, flags bits & opcode
  - Get these in just one pass over the pcaps with **`tcpdump`**

# HOW THE CRUNCHING GOT DONE

- 2 passes over both 2012 and 2013 DITL RSO pcaps
  - First pass counted TLDs using packetq
    - Took about 2 CPU-months (1 week of elapsed time)
- Second pass got **`tcpdump`** to pretty-print packets where QNAME contained a new gTLD label
  - Took nearly 2 CPU-months
  - **`awk`** scripts took about 1 CPU-week to analyse label positions, count source address prefixes

# AN UGLY GOTCHA

- Some **`packetq`** output from 2013 data was wrong:
  - Essentially null files were produced
- Duane Wessels explained some raw pcaps had used 802.1q VLAN tagging
  - Hadn't been tidied up at OARC at that point
- **`packetq`** treated 802.1q link-level header as payload
  - An off by 4 bytes error...
  - Henrik Levkowetz at Netnod fixed this very quickly :-)

# FINDINGS - 1

- 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

# FINDINGS - 2

- Very, very, very long tail
  - Several billion 10-character TLDs seen exactly once
  - Looks to be NXDOMAIN detection/mitigation by **`chrome`**
- Ran out of time to look at flag bits and opcodes
  - Sampling didn't find lots of non-query opcodes
  - Unsure of the share of queries from proper resolvers and stub resolvers or forwarding-only boxes

# 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?
- Should be looked at in more detail

# LESSONS LEARNED

- The old tools from the 80s should not be overlooked
  - They can still do useful stuff
  - Fewer unknowns & dependencies, especially on an old OS
- Use simple tools that do just one thing and do them well
  - Generally better than the Swiss army penknife approach that **`perl`**/**`python`**/***`whatever`*** seem to typify
- Always remember Ken Thompson's advice: "when in doubt, use brute force"

# ACKNOWLEDGEMENTS AND THANKS


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

# QUESTIONS?