Building OpenDNS Stats

Richard Crowley
richard@opendns.com
Then: 8 billion DNS queries per day

Now: 14 billion DNS queries per day
Logs are silly, let’s make graphs
map/reduce/ish

Stage 1 buckets data by network
Stage 2 aggregates and stores

Prefers to duplicate data rather than omit data

Give each network a separate table (keeps each table small(er) and keeps the primary key small(er))
False start #1: storing domains

auto_increment is bad (table lock)

Use the SHA1 of the domain as primary key

Currently we have 2 machines storing domains
About 48 GB in each domains.ibd
28 GB memcached across 8 machines effectively makes this database write-only
Stage 2 aggregated too much data and ran out of memory

Bad idea: improve the heuristic used to guess memory usage and prevent `std::bad_alloc`

Good idea: catch `std::bad_alloc`, clean up and restart

Pre-allocating buffers that will be reused makes this easy

Protip: Run two programs (`memcached` and Stage 2, for example) compiled 32-bit on a 64-bit CPU with 8 GB RAM
False start #3: open tables

80+ %iowait from opening and closing tables

strace showed lots of calls to open() and close()

strace crashed MySQL

Altered mysqld_safe to set ulimit -n 600000
False start #4: MyISAM

Didn’t mind table locks, so I used MyISAM

12 MB/sec total across 4 nodes

Migration to InnoDB is in progress

Expect a 2x improvement from InnoDB

innodb_flush_log_at_trx_commit=2
Architecture
Bird’s eye view

Web servers (Palo Alto)

Stats DBs

Stage 2

Stage 1

San Francisco

Domains DB

User DB

Resolvers (worldwide)
**Stage 1 ("map")**

`rsync` log files from our DNS servers to 3 servers in San Francisco

Looking up a network in `memcached` (or `$GLOBALS`) gives the preferred Stage 2

Write log lines back to local disk, one bucket for each Stage 2 machine

Future work: automated rebalancing and failover
Stage 2 data structures

Stats aggregation (pseudocode)

```
{  
  "db1": {  
    "123456": {  
      "2009-06-17": {  
        "last_updated": 1234567890,  
        "file_ptrs": [0xDEADBEEF, 0xDECAFBAD],  
        "topdomains": {  
          "xkcd.com": [12,3,5,47,0,0,6,10,1,9,2,3,0,4,2,0,5,12,19,35,32,2,4,0],  
        },  
        "requesttypes": { "A": [ /* 24 hours */ ], "MX": [ /* 24 hours */ ] },  
        "uniqueips": { "1.2.3.4": [ /* 24 hours */ ] }  
      }  
    }  
  }  
}
```

File reference counting (C++)

```
__gnu_cxx::hash_map<
  char *, // Filename  
  std::pair<  
    unsigned int, // Reference count  
    pthread_t // Owning thread or NULL  
  >,  
  hash_ptr // Hashes a pointer as if it were an integer
>
```
Stage 2 ("reduce")

rsync intermediate files from all Stage 1 servers

8 aggregator threads read intermediate files into memory

8 pruning threads write SQL statements to disk
They decide what to prune based on the last_updated time
They prefer to prune data that allows many files to be deleted

Files are reference counted and only deleted when all of their rows are on disk as SQL
Stats Databases (“satan”)

MySQL 5.0.77-percona
12 disks
16 GB RAM

`table_cache=300000`

`innodb_dict_size_limit=2G`
`innodb_flush_log_at_trx_commit=2`
opendns.com is in Palo Alto
DNS Stats are in San Francisco

(Private) JSON API proxies small chunks of stats data to the website as needed

Queries are done with no LIMIT clause
Results are paginated in memcached (TTL = 1 hour)
Questions?

http://opendns.com/dashboard/stats

http://rcrowley.org/talks/opendns_stats.pdf

richard@opendns.com