A Distributed Multicast DNS System

Daniel Johnson

June 15, 2010

Purpose and Scope

To allow a group of workstations as present on a medium-size subnet to survive complete loss of a DNS server through collaboration. The syslab workstations provide an example of an appropriate size.

Background Research

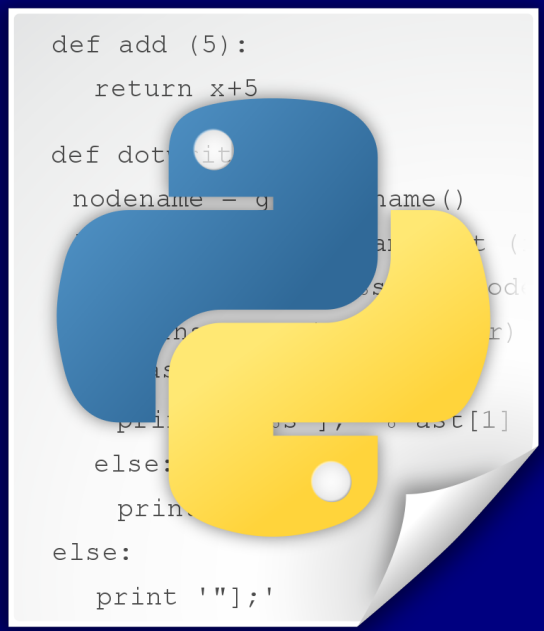
Several efforts have been made to solve similar problems:

- The Avahi project provides name lookups for machines on the local subnet vial a .local pseudo-TLD
- DistributedDNS attempts to surpass the traditional ICANN-based name service, which is too ambitious to succeed.

My solution with work with the local link only, which will keep speed as fast as or faster than traditional nameservers, provide lookups for all hosts, not just nearby ones, and honor the authority of the root nameservers.
Another important algorithm may be DHT: Distributed Hash Tables. Since DNS records are key-value pairs, this may be perfect. However, DHT seems to work better for larger sized things, it is difficult to determine how it can be used for much smaller things like DNS records.

Computer Language/Software

Python is being used for prototyping and network simulation.



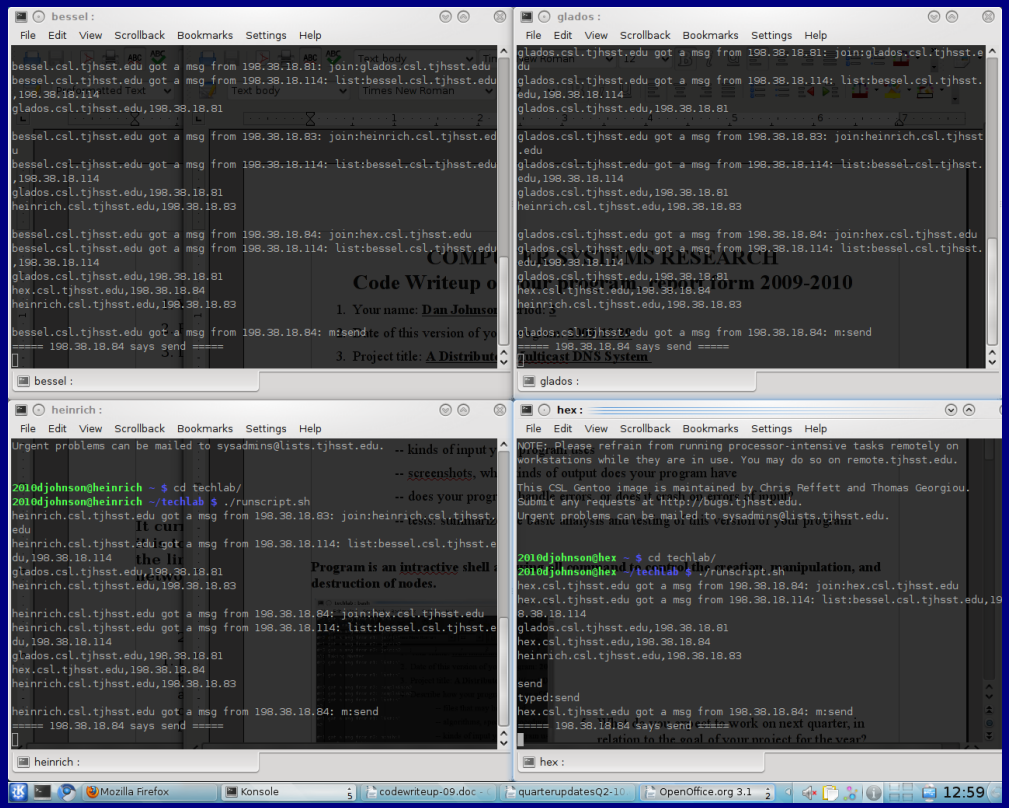
Procedure

- Python
  - Proof-of-concept
  - Protocol development
  - Multicast development

Testing/Analysis

- Simulation tests
  - Network chatter
  - Protocol development
  - Degradation of network

Example



An example run of the simulation.

Expected Results

The benefits of offloading routine and emergency duties from the nameserver has several practical benefits. First, in the event of a nameserver outage, not all systems need to fail. While non-cached entries may not be available, those that have seen high use (google.com, for example) will still be available. This helps to eliminate one instance of a single point of failure. With a sufficient number of hosts, processing queries on the main nameserver can lead to performance issues. By dividing responsability for name lookups among hosts, the speed and scalability of lookups can be improved.

Progress Completed

The proof of concept functions as expected. The hybrid client-servers connect to each other, retain only the information they are interested in, and communicate these things clearly enough to help each other out.

Possible Future Work

Two things need further work to extend this project past the proof-of-concept stage: integration with system applications and a better method of visualizing the information from the hosts. Currently, the only way to look at the output of a handful of nodes is by having them all open in different tabs of a terminal, which makes viewing more than a couple of nodes worth of output difficult. A graphical view will make testing and debugging easier, as well as provide a better visualization method for showing the technology.

It is my hope that further work can be done on this to extend my proof of concept into somthing that can be deployed to labs in real schools and businesses.