Domain Name System (DNS)

Ayitey Bulley
abulley@ghana.com

Objectives

- Describe why we need name to address mappings and what was wrong with HOSTS.TXT
- Describe the client-server model of DNS
- Setup and test a resolver on a UNIX machine
- List and describe resource records (RR) a client would be interested in
- Use dig tool to resolve names with more detail

Why Names?

- The Internet infrastructure depends on IP addresses
- Machines communicate with each other via IP addresses
- However human beings can remember names better than numbers
- A single file mapping name to IP address was created (HOSTS.TXT)

HOSTS.TXT

- Maintained by SRI's NIC and distributed from a single host
- Administrators emailed changes to the NIC and periodically downloaded the current HOSTS.TXT
- Changes were compiled into a new HOSTS.TXT once or twice a week
- As the network grew this scheme became impracticable.

What was wrong with HOSTS.TXT

- Traffic and load
- Name collisions
- Consistency
- Single point of editing and maintenance
- Did not scale well
- The need for a more scalable system/scheme, hence DNS

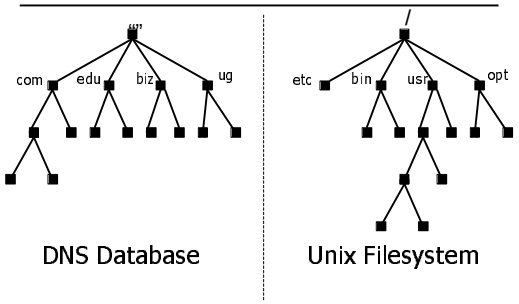
What is DNS?

- DNS is a distributed database
- Allows local control of segments of the overall database
- Employs a client-server architecture
- Robustness and performance achieved through replication and caching
- Name servers constitute the server half of the client-server mechanism
- Resolvers constitute the client half of the client-server mechanism
- Structure of the DNS database is hierarchical

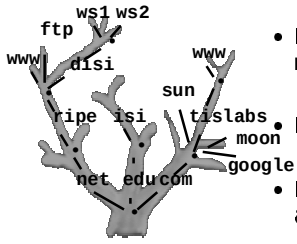
Hierarchical Structure of DNS

- Very similar to the structure of the UNIX file system
- Pictured as an inverted tree with root node at the top
- Each node in the tree has a text label
- The null label "" is reserved for the root node
- Root node is written as a single dot (.)

Hierarchical Structure of DNS (contd.)



Hierarchical Structure of DNS (contd.)



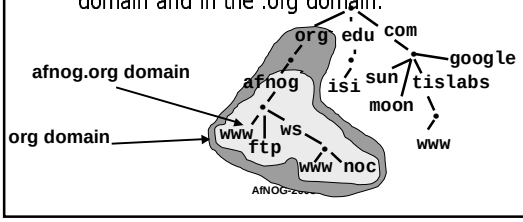
- Domain names can be mapped to a tree.
- New branches at the 'dots'
- No restriction to the amount of branches.

Hierarchical Structure of DNS (contd.)

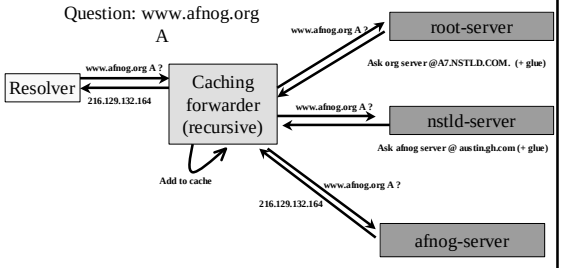
- Hostnames are globally unique
  - E.g. pc1.t1.ws.afnog.org and pc1.t2.ws.afnog.org
- Name space is administered in zones
  - E.g. afnog.org and ws.afnog.org can be administered by different organizations

Domains

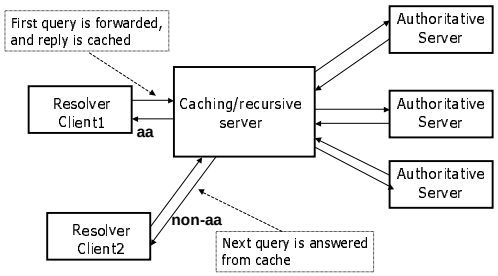
- Domains are "namespaces"
- Everything below .com is in the com domain.
- Everything below afnog.org is in the afnog.org domain and in the .org domain.



Client-Server Model



Client-Server Model



Types of Nameservers

- Caching-Only Server
  - Non-authoritative for any zone except 0.0.127.in-addr.arpa
  - Resolves recursively by querying authoritative nameservers
- Authoritative Servers
  - Master (Primary)
  - Slave (Secondary)
    - Can be authoritative for one or more domains

Exercise 1

- In this exercise we will be testing name resolution using:
  - ping and a browser
  - and by configuring the local resolver on your PCs (/etc/resolv.conf)
- Please refer to Exercise 1 in the handouts given to you.

Client Resolver Lookups

- Possible queries from a client to a nameserver are:
  - Name to IP Address (A)
    - browser
  - Name to Mail exchanger (MX)
    - Mail Server (MTA) e.g. Exim
  - IP Address to Name (PTR) - [Reverse DNS]
    - Logging of incoming connections (apache logs)
  - Alias to Name (CNAME)
  - Other resource record (RR) types
    - SOA, NS (Mainly Server to Server)

Client Resolver Lookups (contd.)

- Possible responses from a nameserver to a client are:
  - Positive answer
  - Negative answer (Name does not exist)
  - Server Fail (Could not find any answer)

Client Utilities for Testing DNS

- BIND comes with utilities for testing and troubleshooting nameserver issues. Some of these tools are:
  - nslookup
  - dig
- Most client programs use the local resolver
  - E.g. ping, browsers etc.
- In this workshop we will focus on the dig and ping utilities.

The BIND dig utility

- Syntax

dig [@server] domain [q-type] [other options]

- Server – The server you want to use to resolve the query (defaults to servers listed in /etc/resolv.conf)
- Domain - a name in the Domain Name System
- q-type - is one of (a,any,mx,ns,soa,hinfo,axfr,txt,…) [default: a]

- Examples

```
# dig @81.199.109.1 ws.afnog.org a
# dig @ns.t1.ws.afnog.org ws.afnog.org a
# dig @noc.ws.afnog.org -x 81.199.110.100
# man dig
```


Question

- From the output from the last example, what is the default query type?

Understanding output from dig

- Queries using the dig utility outputs a lot of information, however the most important for us are
  - Status
  - Flags
  - Answer Section
  - Authority Section
  - Additional Section
  - TTL
  - Total query time
  - "From …. To …. Server" …. Section


```
ns# dig @81.199.110.100 www.gouv.bj a

; <<>> DiG 8.3 <<>> @81.199.110.100 www.gouv.bj a
; (1 server found)
;; res options: init recurs defnam dnsrch
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR  id: 4
;; flags: qr aa rd ra;  QUERY: 1, ANSWER: 2, AUTHORITY: 4, ADDITIONAL: 3
;; QUERY SECTION:
;;      www.gouv.bj, type = A, class = IN

;; ANSWER SECTION:
www.gouv.bj.            1D IN CNAME     waib.gouv.bj.
waib.gouv.bj.           1D IN A         208.164.179.196

;; AUTHORITY SECTION:
gouv.bj.                1D IN NS        rip.psg.com.
gouv.bj.                1D IN NS        ben02.gouv.bj.
gouv.bj.                1D IN NS        nakayo.leland.bj.
gouv.bj.                1D IN NS        ns1.intnet.bj.

;; ADDITIONAL SECTION:
ben02.gouv.bj.          1D IN A         208.164.179.193
nakayo.leland.bj.       1d23h59m59s IN A  208.164.176.1
ns1.intnet.bj.          1d23h59m59s IN A  81.91.225.18

;; Total query time: 2084 msec
;; FROM: ns.t1.ws.afnog.org to SERVER: 81.199.110.100  81.199.110.100
;; WHEN: Sun Jun  8 21:18:18 2003
;; MSG SIZE  sent: 29  rcvd: 221
```


Exercise 2

- In this exercise we will be using the dig utility to resolve domain names
  - dig using your local resolver
  - dig using another caching server
  - dig for reverse lookups
  - dig for a non-existent domain

Best Practices

- Choose caching nameservers close by you for your resolver
- Select at least two (2) caching nameservers for your resolver (redundancy)
- Use search lists in the resolver for non-FQDN