IP ADDRESSES, NAMING, AND DNS

George Porter
Apr 9, 2018

ATTRIBUTION
• These slides are released under an Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0) Creative Commons license
• These slides incorporate material from:
  • Computer Networks: A Systems Approach, 5e, by Peterson and Davie
ANNOUNCEMENTS

Reading due:
Today: Donahoo and Calvert, Chapter 3
Wednesday: Donahoo and Calvert, Chapter 2

Outline

1. Internetworking overview
2. IP and IP addresses
3. DNS and naming
4. DNS API
INTERNETWORKING

• What is an internetwork?

• An arbitrary collection of networks interconnected to provide some sort of host-host to packet delivery service

INTERNETWORKING PROTOCOL

• Each host has a “local” address on specific sub-network

• Ethernet, WiFi, UCSD, Comcast, AT&T, Verizon Wireless, ...

• Yet each host has a single, global “IP” address
Outline

1. Internetworking overview
2. IP and IP addresses
3. DNS and naming
4. DNS API

IP SERVICE MODEL

- Packet Delivery Model
  - Connectionless model for data delivery
  - Best-effort delivery (unreliable service)
    - packets are lost
    - packets are delivered out of order
    - duplicate copies of a packet are delivered
    - packets can be delayed for a long time
- Global Addressing Scheme
  - Provides a way to identify all hosts in the network
**IP PACKET FORMAT**

- Version (4): currently 4
- Hlen (4): number of 32-bit words in header
- TOS (8): type of service (not widely used)
- Length (16): number of bytes in this datagram
- Ident (16): used by fragmentation
- Flags/Offset (16): used by fragmentation
- TTL (8): number of hops this datagram has traveled
- Protocol (8): demux key (TCP=6, UDP=17)
- Checksum (16): of the header only
- DestAddr & SrcAddr (32)

**GLOBAL ADDRESSES**

- Properties
  - globally unique
  - hierarchical: network + host
  - 4 Billion IP address, half are A type, ¼ is B type, and 1/8 is C type
- Format
- Dot notation
  - 10.3.2.4
  - 128.96.33.81
  - 192.12.69.77
CIDR: CLASSLESS INTERDOMAIN ROUTING

- Original IP address design: limited network sizes
  - 256, 65536, or 16777216 hosts per network
    - Not very flexible!
- CIDR enables any power-of-two network size
  - Networks with 16 hosts, or 32 hosts, etc.
  - Number of bits assigned to the host part of the address indicated with a “/”

CIDR EXAMPLES

- 192.168.1.1/16
  - First 16 bits = network, last 16 bits = $2^{16}$ hosts
- 206.109.3.1/24
  - First 24 bits = network, last 8 = 256 hosts
- 212.110.9.1/30
  - First 30 bits = network, only 4 hosts in that network!
ASSIGNING ADDRESSES VIA DHCP

- DHCP server is responsible for providing configuration information to hosts
- There is at least one DHCP server for an administrative domain
- DHCP server maintains a pool of available addresses

DHCP IN ACTION

- Newly booted or attached host sends DHCPDISCOVER message to a special IP address (255.255.255.255)
- DHCP relay agent unicasts the message to DHCP server and waits for the response
Outline

1. Internetworking overview
2. IP and IP addresses
3. DNS and naming
4. DNS API

DNS HOSTNAME VERSUS IP ADDRESS

• **DNS host name** (e.g. www.cs.ucsd.edu)
  • **Mnemonic** name appreciated by humans
  • **Variable length**, full alphabet of characters
  • Provides **little** (if any) information about **location**

• **IP address** (e.g. 128.112.136.35)
  • Numerical address appreciated by **routers**
  • **Fixed length**, decimal number
  • **Hierarchical** address space, related to host **location**
MANY USES OF DNS

• Hostname to IP address translation
• IP address to hostname translation (reverse lookup)
• Host name aliasing: other DNS names for a host
  • Alias host names point to canonical hostname
• Email: Lookup domain’s mail server by domain name

ORIGINAL DESIGN OF DNS

• Per-host file named /etc/hosts (1982)
• Flat namespace: each line = IP address & DNS name
• SRI (Menlo Park, California) kept the master copy
• Everyone else downloads regularly
• But, a single server doesn’t scale
  • Traffic implosion (lookups and updates)
  • Single point of failure
• Need a distributed, hierarchical collection of servers
DNS: GOALS AND NON-GOALS

- A wide-area **distributed database**
- Goals:
  - **Scalability**; decentralized maintenance
  - **Robustness**
  - Global scope
    - Names mean the same thing everywhere
  - Distributed updates/queries
  - Good **performance**
- But don’t need strong consistency properties

DOMAIN NAME SYSTEM (DNS)

- Hierarchical name space divided into contiguous sections called **zones**
- Zones are distributed over a collection of DNS servers
- Hierarchy of DNS servers:
  - **Root** servers (identity hardwired into other servers)
  - **Top-level domain (TLD)** servers
  - **Authoritative** DNS servers
- Performing the translations:
  - **Local DNS servers** located near clients
  - **Resolver** software running on clients
DNS IS HIERARCHICAL

- Hierarchy of namespace matches hierarchy of servers
- Set of namerservers answers queries for names within zone
- Nameservers store names and links to other servers in tree

DNS ROOT NAMESERVERS

- 13 root servers. Does this scale?
DNS ROOT NAMESERVERS

- 13 root servers. *Does this scale?*
- Each server is really a *cluster of servers (some geographically distributed), replicated via IP anycast*
LOCAL NAME SERVERS

- Do not strictly belong to hierarchy
- Each ISP (or company, or university) has one
  - Also called default or caching name server
- When host makes DNS query, query is sent to its local DNS server
  - Acts as proxy, forwards query into hierarchy
  - Does work for the client

DNS RESOURCE RECORDS

- DNS is a distributed database storing resource records
- Resource record includes: (name, type, value, time-to-live)

  Type = A (address)
  - name = hostname
  - value is IP address

  Type = CNAME
  - name = alias for some “canonical” (real) name
  - value is canonical name

  Type = NS (name server)
  - name = domain (e.g. princeton.edu)
  - value is hostname of authoritative name server for this domain

  Type = MX (mail exchange)
  - name = domain
  - value is name of mail server for that domain
DNS IN OPERATION

- Most queries and responses are UDP datagrams
- Two types of queries:
  - **Recursive**: Nameserver responds with answer or error
    
    ![Recursive DNS Diagram](image1)
    
    **Answer**: www.ucsd.edu A 132.239.180.101
  - **Iterative**: Nameserver may respond with a referral
    
    ![Iterative DNS Diagram](image2)
    
    **Referral**: .edu NS a.edu-servers.net.

RECURSIVE DNS IN ACTION

![Recursive DNS Diagram](image3)
RECURSIVE VERSUS ITERATIVE QUERIES

Recursive query

• Less burden on entity initiating the query
• More burden on nameserver (has to return an answer to the query)
• Most root and TLD servers won’t answer (shed load)
  • Local name server answers recursive query

Iterative query

• More burden on query initiator
• Less burden on nameserver (simply refers the query to another server)

DNS CACHING

• Performing all these queries takes time
  • And all this before actual communication takes place
• Caching can greatly reduce overhead
  • The top-level servers very rarely change
    • Popular sites visited often
  • Local DNS server often has the information cached
• How DNS caching works
  • All DNS servers cache responses to queries
  • Responses include a time-to-live (TTL) field
    • Server deletes cached entry after TTL expires
Outline

1. Internetworking overview
2. IP and IP addresses
3. DNS and naming
4. DNS API

MAPPING NAMES TO ADDRESSES

GETADDRINFO(3) Linux Programmer's Manual GETADDRINFO(3)

NAME
getaddrinfo, freeaddrinfo, gai_strerror — network address and service translation

SYNOPSIS
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>

int getaddrinfo(const char *node, const char *service,
    const struct addrinfo *hints,
    struct addrinfo **res);

void freeaddrinfo(struct addrinfo *res);

const char *gai_strerror(int errcode);
LINKED LIST OF ‘ADDRINFO’ STRUCTS

```c
struct addrinfo {
    int ai_flags;
    int ai_family;
    int ai_socktype;
    int ai_protocol;
    socklen_t ai_addrlen;
    struct sockaddr *ai_addr;
    char *ai_canonname;
    struct addrinfo *ai_next;
};
```

- Q: Why a linked list?
- Q: Which of the multiple results should you use?

ANNOUNCEMENTS

Reading due:
- Today: Donahoo and Calvert, Chapter 3
- Wednesday: Donahoo and Calvert, Chapter 2
UC San Diego