CSE/EE 461

Ratul Mahajan
ratul@cs.washington.edu

19th Nov, 2003
Inter-Domain Routing (BGP)

- Network comprised of many Autonomous Systems (ASes)
- To scale, use hierarchy: separate inter-domain and intra-domain routing
- Run a path vector protocol between ASes
Path Vectors

- Similar to distance vector, except send entire paths
  - e.g. 321 hears [7,12,44]
  - stronger avoidance of loops
  - supports policies

- Modulo policy, shorter AS-paths are preferred
  - Reachability only – no metrics
Routing Policies

- Selecting non-default routes
  - performance, cost, business considerations
- Local policy dictates what route will be chosen and what routes will be advertised!
  - e.g., X doesn’t provide transit for B, or A prefers not to use X
Providers and Peers

◆ Sending traffic
  • customers (most preferred)
  • peers
  • providers (least preferred)

◆ Providing transit
  • do not provide transit for peers

◆ To peer or not to peer?
Routing policies considered harmful

◆ Policy is embedded in routing advertisement and route selection
  • known only to the ISP
◆ Too many knobs in BGP
  • hard to get it right
◆ When a route changes
  • is it policy or is it failure?
◆ Routing may never converge
  • conflicting policies
◆ Hard to predict the impact of local changes
  • leads to instabilities
Multiple peering points (policy)

- Usually between tier-1 ISPs
  - bandwidth
  - latency
- Peering point selection
  - Early Exit / Hot Potato
  - Late Exit / Cold Potato
- Side-effect: asymmetry
- Combination of best local policies not globally best
Does BGP make Internet routing evolvable?

- Yes
  - what happens inside domains does not matter

- No
  - cannot change BGP itself (is a big problem today)
  - Options:
    - implement changes under BGP
    - implement changes above BGP
Inter-domain routing (recap)

 Goals:
 - scalability
 - policies

 Approach:
 - hierarchy, information hiding, flexible path selection and export
Addressing

❖ Scalability
  • address space exhaustion
  • routing table size
    ▪ $2^{32}$ (4B) addresses

❖ First cut: partition the space into 3 classes
  • A (24 bits for host addresses) + B (16) + C (8)
  • Too few class B instances
  • Can give multiple class C addresses, but ...

❖ Solution: flexible address space boundaries
CIDR (Supernetting)

- Classless Inter-Domain Routing
- Aggregate advertised network routes
  - represent as prefix/bits
- Reduces routing table size
- Requires sensible address assignment
CIDR Example

- X and Y routes can be aggregated

Border gateway (advertises path to 110000000000001) → Regional network →
Corporation X (1100000000000001000001)
Corporation Y (1100000000000001000000)
CIDR issues

- Problems with multihoming
- Complicates routing table lookup
  - requires longest prefix match
Addressing (recap)

- Goal: scalability

- Approach:
  - route aggregation and careful address allocation
Naming

- How do we name objects in the Internet?
- How do we find them?
- Scalability is a key goal (again!)
**Names vs. Addresses**

- **Names** are identifiers for objects/services (high level)
- **Addresses** are locators for objects/services (lower level)
- **Resolution** is the process of mapping names to addresses
- But addresses are really lower-level names
Naming in Systems

- Ubiquitous
  - Files in filesystem, processes in OS, pages on the web, ...

- Decouple identifier for object/service from location
  - Hostnames provide a level of indirection for IP addresses

- Naming greatly impacts system capabilities and performance
  - Ethernet addresses are a unique flat 48 bits
    - flat → any address anywhere but large forwarding tables
  - IP addresses are hierarchical 32/128 bits
    - hierarchy → smaller routing tables but constrained locations
Internet Hostnames

- Hostnames are human-readable identifiers for end-systems based on an administrative hierarchy
  - galah.cs.washington.edu is my desktop machine
- IP addresses are a fixed-length binary encoding for end-systems based on their position in the network
  - 128.95.2.106 is galah’s IP address

- Original name resolution: HOSTS.TXT
- Current name resolution: Domain Name System
- Future name resolution: ?
Original Hostname System

- When the Internet was really young ...

- Flat namespace
  - Simple (host, address) pairs

- Centralized management
  - Updates via a single master file called HOSTS.TXT
  - Manually coordinated by the Network Information Center (NIC)

- Resolution process
  - Look up hostname in the HOSTS.TXT file
Scaling Problems

◆ Coordination
  • Between all users to avoid conflicts

◆ Inconsistencies
  • Between update and distribution of new version

◆ Reliability
  • Single point of failure

◆ Performance
  • Competition for centralized resources
Domain Name System (DNS)

- Namespace is hierarchical
  - Allows much better scaling of data structures
  - e.g., galah.cs.washington.edu

- Namespace is distributed
  - Decentralized administration and access
  - e.g., galah managed by CSE

- Resolution is by query/response
  - With replicated servers for redundancy
  - With heavy use of caching for performance
“dot” is the root of the hierarchy
- Top levels now controlled by ICANN
- Lower level control is delegated
- Usage governed by conventions
- FQDN = Fully Qualified Domain Name
DNS Components

- Data managed by **zones** that contain **resource records**
  - Zone is a complete description of a portion of the namespace
  - e.g., all hosts and addresses for machines in washington.edu with pointers to subdomains like cs.washington.edu

- One or more **nameservers** manage each zone
  - Zone transfers performed between nameservers for consistency
  - Multiple nameservers provide redundancy

- Clients query nameservers for specified records
  - Multiple messages may be exchanged per DNS lookup to navigate the name hierarchy (coming soon)
Partial results (e.g. server for princeton.edu) are cached too
Hierarchy of Nameservers

Root name server

Princeton name server

CS name server

Cisco name server

EE name server
Availability

- What happens if some name server is not working?
- Replicate name servers
  - resolution service is available if at least one replica is working
  - queries can also load balance between replicas
Caching

- Servers and clients cache results of DNS lookups
  - Cache partial results too (e.g., server for princeton.edu)
  - Greatly improves system performance; lookups the rare case

- Cache using time-to-live (TTL) value from provider
  - higher TTL means less traffic, lower TTL means less stale info

- Negative caching is used too!
  - errors can cause repeated queries for non-existent data
DNS Bootstrapping

- Need to know IP addresses of root servers before we can make any queries

- Addresses for 13 root servers ([a-m].root-servers.net) handled via initial configuration (named.ca file)
Building on the DNS

- Other naming designs leverage the DNS

- Email:
  - ratul@cs.washington.edu is ratul in domain cs.washington.edu
  - uses a separate record type to find the mail server

- Uniform Resource Locators (URLs) name for Web pages
  - e.g., www.cs.washington.edu/homes/djw
  - Use domain name to identify a Web server
  - Use “/” separated string to name path to page (like files)
DNS recap

- The design of names, addresses and resolution has a significant impact on system capabilities

- Hierarchy, decentralization and caching allow the DNS to scale
  - These are general techniques!