Last Time

- SOOTS today; please remind me 15 mins early
- Last time – QoS
  - QoS (explicit resource allocation) vs. congestion management
  - Weighted fair queueing and Leaky bucket
  - The Contract: Need to specify the application requirements are
  - Intserv vs. Diffserv
  - Supporting IntServ

Describing the Flow – Leaky Bucket

- Bucket leaks at a rate of $r$ packets per unit time
- Bucket can hold $B$ packets
- If packets arrive in a way that overfills the bucket, they are dropped
- Apply it to the flow examples above
- $r$ describes the average; $B$ the variability

Example – Multiple Flows

- Virtual clock is advanced 1 tick for every $n$ bits transmitted if there are $n$ active flows
- Calculate $F_i$ for each packet as it arrives on each flow
- Send packet with minimum $F_i$
- Cannot preempt a packet that is already in transit
- Think about the ATM cell size selection
- WFQ: give a flow a higher weight by making its clock tick slower

Reference Implementation (RFC 1633)

- Classifier
- Packet Scheduler
- Admission Control
- Reservation Setup Protocol (RSVP)
Flowspecs

- Flowspecs: the user’s definition of the service required by the flow
- Consists of two decoupled specifications
  - Rspec: describes the service required from the network (e.g., predictive or guaranteed delay/delay target)
  - Tspec describes the flow’s traffic characteristics
    - Leaky bucket used for Tspec
    - What are the implications on the resources needed?

Mechanisms – Implementing the reservation

- Two reservation messages: PATH and RESV
- A soft state is maintained by participating routers when the RSVP reply (RESV) packet is received
- What happens if the route changes?
- State is leased and must be renewed every 30 seconds
- Non participating routers are still best effort
- Support for Multicast QoS – will not go into it

Connectionless Flows

- Even though IP is connectionless, usually packets follow the same route
- Can associate a “soft state” with flows to allow connection-oriented handling
- With connection-oriented, there is a hard-state associated with the flow

Resource Setup Protocol

- Resource ReSerVation Protocol (RSVP)
- Uses flowspecs to describe flows
- Sender sends flow specification to receivers
- Receivers respond with reservation request
  - May reserve less than flow spec (e.g., low-quality audio is sufficient even though high-quality is transmitted)
  - May reserve more than flow spec (to receive from multiple senders)
  - May reserve less than flow spec (to receive low-quality audio is sufficient even though high-quality is transmitted)
- Request travels back to the sender along the path reserve appropriate bandwidth and buffers
- Routers merge reservation paths
Differentiated Services

Quality of service associated with aggregates instead of a single flow

Routers can then deal with these aggregates differently according to their service requirements

Example: IP's expedited forwarding option (EF) – EF packets placed in separate queue

Police to some maximum peak rate, and drop offending packets if they exceed

Is it practical to guess the shape of the aggregated traffic?

Much more scalable than IntServ

What happens to flows within an aggregate: will they be able to meet their service requirements?

Many more scalable than IntServ

Example DiffServ: Controlled Load

Service guarantee: network looks “lightly loaded” for conforming traffic

Assured Forwarding (RFC 2597)

- Four Independent traffic classes
- Three drop preference levels within each class

Policing: Police to a specified rate and burst profile

- Mark out of profile packets to have higher drop probability

Properties

- Names vs. Addresses
- Location transparent vs. location-dependent
- Flat vs. Hierarchical
- Global vs. local
- Absolute vs. relative
- By architecture vs. by convention
- Unique vs. ambiguous

Naming

Overview
- What do names do?
  - Identify objects
  - Help locate objects
  - Define membership in a group
  - Specify a role
- Directory services
  - A set of Name to value bindings
  - What are they used for?
  - Yellow pages vs. white pages

Identify objects

Help locate objects

Define membership in a group

Specify a role

A set of Name to value bindings

What are they used for?

Yellow pages vs. white pages

What are they used for?

Yellow pages vs. white pages

Properties

Names vs. Addresses

Location transparent vs. location-dependent

Flat vs. Hierarchical

Global vs. local

Absolute vs. relative

By architecture vs. by convention

Unique vs. ambiguous

Naming

Overview
- What do names do?
  - Identify objects
  - Help locate objects
  - Define membership in a group
  - Specify a role
- Directory services
  - A set of Name to value bindings
  - What are they used for?
  - Yellow pages vs. white pages

Identify objects

Help locate objects

Define membership in a group

Specify a role

A set of Name to value bindings

What are they used for?

Yellow pages vs. white pages

What are they used for?

Yellow pages vs. white pages

Properties

Names vs. Addresses

Location transparent vs. location-dependent

Flat vs. Hierarchical

Global vs. local

Absolute vs. relative

By architecture vs. by convention

Unique vs. ambiguous

Naming

Overview
- What do names do?
  - Identify objects
  - Help locate objects
  - Define membership in a group
  - Specify a role
- Directory services
  - A set of Name to value bindings
  - What are they used for?
  - Yellow pages vs. white pages

Identify objects

Help locate objects

Define membership in a group

Specify a role

A set of Name to value bindings

What are they used for?

Yellow pages vs. white pages

What are they used for?

Yellow pages vs. white pages

Properties

Names vs. Addresses

Location transparent vs. location-dependent

Flat vs. Hierarchical

Global vs. local

Absolute vs. relative

By architecture vs. by convention

Unique vs. ambiguous

Naming

Overview
- What do names do?
  - Identify objects
  - Help locate objects
  - Define membership in a group
  - Specify a role
- Directory services
  - A set of Name to value bindings
  - What are they used for?
  - Yellow pages vs. white pages

Identify objects

Help locate objects

Define membership in a group

Specify a role

A set of Name to value bindings

What are they used for?

Yellow pages vs. white pages

What are they used for?

Yellow pages vs. white pages

Properties

Names vs. Addresses

Location transparent vs. location-dependent

Flat vs. Hierarchical

Global vs. local

Absolute vs. relative

By architecture vs. by convention

Unique vs. ambiguous

Naming

Overview
- What do names do?
  - Identify objects
  - Help locate objects
  - Define membership in a group
  - Specify a role
- Directory services
  - A set of Name to value bindings
  - What are they used for?
  - Yellow pages vs. white pages

Identify objects

Help locate objects

Define membership in a group

Specify a role

A set of Name to value bindings

What are they used for?

Yellow pages vs. white pages

What are they used for?

Yellow pages vs. white pages

Properties

Names vs. Addresses

Location transparent vs. location-dependent

Flat vs. Hierarchical

Global vs. local

Absolute vs. relative

By architecture vs. by convention

Unique vs. ambiguous

Naming

Overview
- What do names do?
  - Identify objects
  - Help locate objects
  - Define membership in a group
  - Specify a role
- Directory services
  - A set of Name to value bindings
  - What are they used for?
  - Yellow pages vs. white pages

Identify objects

Help locate objects

Define membership in a group

Specify a role

A set of Name to value bindings

What are they used for?

Yellow pages vs. white pages

What are they used for?

Yellow pages vs. white pages

Properties

Names vs. Addresses

Location transparent vs. location-dependent

Flat vs. Hierarchical

Global vs. local

Absolute vs. relative

By architecture vs. by convention

Unique vs. ambiguous

Naming

Overview
- What do names do?
  - Identify objects
  - Help locate objects
  - Define membership in a group
  - Specify a role
- Directory services
  - A set of Name to value bindings
  - What are they used for?
  - Yellow pages vs. white pages

Identify objects

Help locate objects

Define membership in a group

Specify a role

A set of Name to value bindings

What are they used for?

Yellow pages vs. white pages

What are they used for?
Examples

- **Hosts**
  - `opal.cs.binghamton.edu` --> `128.226.123.101`
  - `128.226.123.101` --> `08:00:20:CD:7D:6A`

- **Users**
  - Nael Abu-Ghazaleh --> `nael@cs.binghamton.edu`
  - Nael Abu-Ghazaleh --> `607 777 4748`

- **Files**
  - `/usr/bin/ls` --> (server, field)

- **Services**
  - “Nearby printer with short queue and 2MB”

Domain Name System (DNS)

- **Scalable directory services protocol for the Internet**
- Berkeley Internet Name Domain (bind) on unix machines
- Most common use: directory service to map from host name to IP address

Challenges and Concerns

- **Challenges**
  - How to build a directory system for the whole internet?
  - Can you suggest some approaches (dumb or otherwise)?
    * HOSTS.txt – until mid-1980s
  - Is there a phone directory for the whole world?

- **Concerns/Requirements**
  - Ease of administration
  - Availability
  - Scalability
  - Security
  - Extensibility
Resource Record Types

- A: Value gives the 32-bit IPv4 address
- PTR: value gives hostname for the IP address in the name field
- NS: Value is the name for the host running the name server that knows how to resolve names within the specified domain name
- CNAME: provides canonical name for specified host; used for aliases
- MX: value gives the name for the host running mail server that accepts messages for the specified domain
- Not easily extensible; everyone must agree on changes

MX Example

- When you send mail to nael@opal.cs.binghamton.edu
- Mail program queries DNS for an MX record for opal
- The following info is returned (I used nslookup, querytype=mx):

  opal.cs.binghamton.edu  canonical name = cs.binghamton.edu
  cs.binghamton.edu      preference = 0, mail exchanger = cs.bing.
  cs.binghamton.edu      internet address = 128.226.123.101
  cs.binghamton.edu      name server=bingnet1.cc.binghamton.edu

- These correspond to:
  - (opal.cs.binghamton.edu,cs.binghamton.edu,CNAME,IN)
  - (cs.binghamton.edu,cs.binghamton.edu,MX,IN)
  - (cs.binghamton.edu,128.226.123.101,A,IN)
  - (cs.binghamton.edu,bingnet1...,NS,IN)

Name Server

- Each server maintains a collection of resource records (RR)
- Each record: (Name, Value, Type, Class, TTL)
  - Record indicates binding Name to Value
  - Type specifies the type of binding
  - Class allows other entities to define types
  - TTL: how long the record is valid for
Caching and Replication

- Site-wide cache to speed up resolution of frequently used names
- `gethostbyname()` and `gethostbyaddr()`

Server Hierarchy

- Each zone managed by its own name server
- Should the name server include the full directory?
- Server hierarchy provides scalability and distributed management

Caching and Replication

- 13 root servers (top level servers)
  - Last year, there was a denial of service attack on those
  - What would happen if the attack was successful?
- Caching: to reduce DNS traffic, each resolution is cached locally
  - Recommended TTL for hosts is 2 days
  - Record specifies TTL field (can set it to 0 if no caching is desired)
Getting back to Naming

- DNS provides FQDN to IP name resolution; is this the only type of naming of interest?
- Service Discovery/Intentional Naming: equivalent to yellow pages
  - Find a service — typically not individualized
  - e.g., a printer, or a webserver ...
  - How would you implement it?
  - Would DNS work?

Who are you – reverse DNS

- the domain in-addr.arpa provides reverse mapping
- Used by servers to figure out who is connecting to them
- Records of type PTR
- nslookup with querytype=ptr on 128.226.123.101

```
101.123.226.128.in-addr.arpa name = cs.binghamton.edu
123.226.128.in-addr.arpa nameserver = bingnet1.cc.binghamton.edu
```

- Dynamic DNS

Naming in Peer-to-peer (P2P) Systems

- Networks made up of peers that join and leave arbitrarily and interact with other peers
- Fundamentally different model for applications (compared to client server)
- Peer-to-peer applications can provide
  - redundant storage
  - Permanence (your stuff available when you are disconnected)
  - anycast
  - anonymity
  - etc...

Aside: DNS Cache Poisoning

- Spoofing attack (security next time)
- Mallory (man-in-the-middle) asks its DNS server for the address of hotmail.com
- DNS server recursively sends the request to dns.hotmail.com
- Before it can answer, Mallory spoofs the answer telling the DNS server that hotmail.com is his own machine
- Users now connect to Mallory instead of hotmail.com and volunteer their passwords
- Solution: authenticate DNS replies/updates