Common network/protocol functions

- **Goals:**
  - Identify, study common architectural components, protocol mechanisms
  - *Synthesis:* big picture
  - *Depth:* important topics not covered in introductory courses

- **Overview:**
  - Signaling
  - State
  - Randomization
  - Indirection
  - Network virtualization
  - Multiplexing / Resource Allocation

with slides from Leon-Garcia and Widjaja, Shenker and Stoica
Multiplexing

Multiplexing: *Sharing* resource(s) among users of the resource.

In this lecture:
- The Resources are
  - Bandwidth (Link Capacity)
  - Queues (Buffers)
- The Users are
  - Phone Calls
  - TCP/UDP flows, packets

Other types of resources:
- CPU
- Storage
- Frequency spectrum
... and other types of users?
The resources and the users

H1
H2
H3
H4
H5
H6

R1 output queue
1.5 Mbps bandwidth

R1
R2
Basic facts of life:
- Bandwidth is finite
- Cannot support traffic demands beyond capacity

Example: 1Mbps IP phone, FTP share 1.5 Mbps link
- Bursts of FTP can congest router, cause large delays/audio loss

What’s to be done?
- Move away from the best-effort paradigm
- ... provide “Quality of Service (QoS)”
QoS: What is it?

QoS

Network provides applications with *levels of performance guarantees* needed for applications to function.

*Types of guarantees (service classes)*

- Best-effort (elastic apps)
- Hard real-time (real-time apps)
  - e.g., bounded loss / delay
- Soft real-time (tolerant apps)
  - e.g., probabilistic loss / delay

*How to implement QoS?*

- A set of five principles
Summary of QoS Principles

QoS for networked applications

1. Packet classification
2. Isolation: scheduling and policing
3. High resource utilization
4. Call admission
Principle 1. Traffic/Guarantees specification

- Two 1Mbps IP phones share 1.5 Mbps link
  - want applications to specify: 1) how much bandwidth they need, 2) what levels of guarantees they want

Traffic/guarantees specification (service contract) needed for router to plan whether it can provide certain levels of performance guarantees
Principle 2. Traffic classification

- 1Mbps IP phone, FTP share 1.5 Mbps link
  - bursts of FTP can congest router, cause audio loss
  - want to give priority to audio over FTP
  - Can FTP server declare how much bandwidth it needs?

Packet marking needed for router to distinguish between different classes; and new router policy to treat packets accordingly.
Principle 3. Traffic isolation

- what if applications misbehave (audio sends higher than declared rate)
  - Want to force source adhere to traffic specification

```
Principle 3
provide protection (isolation) for one class from others
```
Principle 4. Call admission

- Bandwidth is finite
  - Cannot support more than available
  - To provide isolation, some flows have to be sacrificed

Principle 4

Call Admission: network may block call (e.g., busy signal) if it cannot meet needs.
Principle 5. Resource sharing

- Allocating *fixed* (non-sharable) bandwidth
  - Inefficient if flows don’t use it
  - Circuit/packet switching; scheduling

While providing isolation, it is desirable to use resources as efficiently as possible.
## Service Classes vs. Principles

<table>
<thead>
<tr>
<th></th>
<th>Traffic / Guarantees Specification</th>
<th>Traffic Classification</th>
<th>Traffic Isolation</th>
<th>Call Admission</th>
<th>Resource Sharing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Best Effort</strong></td>
<td>Yes/No?</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td><strong>Hard real-time</strong></td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td><strong>Soft real-time</strong></td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>
Outline

- Scheduling
- Policing
- Admission Control

- IETF proposals to do things in practice
  - IntServ
  - DiffServ
Scheduling And Policing *Packets*

- **Scheduling**: choose next packet to send on link
- **FIFO (first in first out) scheduling**: send in order of arrival to queue
  - Real-world example: stop sign
- **Discard policy**:
  - Tail drop: drop arriving packet
  - RED
Scheduling Policies: more

Strict Priority scheduling: transmit highest priority queued packet

- Multiple classes, with different priorities
  - class may depend on marking or other header info, e.g. IP source/dest, port numbers, etc.
- real world example: reservations versus walk-ins
Scheduling Policies: still more

Round robin scheduling:

- Multiple classes
- Cyclically scan class queues, serving one from each class (if available)
- Real-world example: 4-way stop (distributed scheduling)
Scheduling Policies: still more

Weighted Fair Queuing:
- generalized Round Robin
- each class gets weighted amount of service in each cycle
Policing Mechanisms

**Goal:** Limit traffic to not exceed declared parameters

Three commonly-used criteria:

- *(Long term) Average Rate:* how many pkts can be sent per unit time (in the long run)
  - crucial question: what is interval length: 100 packets per sec or 6000 packets per min have same average!
- **Peak Rate:** e.g., 6000 pkts per min. (ppm) avg.; 15000 ppm peak rate
- *(Max.) Burst Size:* max. number of pkts sent consecutively (with no intervening idle)
Leaky Bucket Algorithm

Used to police arrival rate + burst size of a packet flow(s)

- Water poured irregularly
- Leak rate corresponds to long-term rate
- Bucket depth corresponds to maximum allowable burst arrival
- Water drains at a constant rate
- 1 packet per unit time
- Assume constant-length packet

Let $X =$ bucket content at last conforming packet arrival
Let $t_a =$ last conforming packet arrival time = depletion in bucket
Policing Mechanisms (more)

- Leaky bucket, WFQ combine to provide guaranteed upper bound on delay, i.e., *QoS guarantee!*

\[
D_{\text{max}} = \frac{b}{R}
\]

arriving traffic

token rate, \( r \)

bucket size, \( b \)

WFQ

per-flow rate, \( R \)

\( r_1 \)

\( b_1 \)

\( r_n \)

\( b_n \)

\( w_1 \)

\( w_n \)
Admission Control

- Users watch either one of two movies
  - Star Wars (with the declared parameters)
    \[ A_1(s, t) \leq \min \{ P_1(t - s), r_1(t - s) + b_1 \} \]
    \[ P_1 = 5 \text{Mbps}, \ r_1 = 2 \text{Mbps}, \ b_1 = .7 \text{Mb} \]
  - Silence of the Lambs (with the declared parameters)
    \[ A_2(s, t) \leq \min \{ P_2(t - s), r_2(t - s) + b_2 \} \]
    \[ P_2 = 4 \text{Mbps}, \ r_2 = 1 \text{Mbps}, \ b_2 = .5 \text{Mb} \]

- Concrete Problem: How many users can be admitted at a C=100Mbps link such that the delay for each user/movie is less than d=200ms?
Admission Control (Contd.)

- Peak rate admission control
  - Provides hard-guarantees

- Dual Leaky-bucket admission control
  - Provides hard-guarantees

- Statistical admission control
  - Provides soft-guarantees, e.g., \( P(Delay > 200ms) \leq 10^{-6} \)

- Average rate admission control
  - Only qualitative guarantees (e.g., delay is always finite)
Admission Control (cont.)

- Peak rate admission control
  - Provides hard-guarantees
  - The formula: $N_1 P_1 + N_2 P_2 \leq C$

- Dual Leaky-bucket admission control
  - Provides hard-guarantees
  - The formula: $N_1 r_1 + N_2 r_2 \leq C, \frac{N_1 b_1 + N_2 b_2}{C} \leq d$

- Statistical admission control
  - Provides soft-guarantees, e.g., $P(\text{Delay} > 200ms) \leq 10^{-6}$

- Average rate admission control
  - Only qualitative guarantees (e.g., delay is always finite)
  - The formula: $N_1 r_1 + N_2 r_2 \leq C$
QoS in the Internet. Part I. IETF Integrated Services

- Architecture for providing QoS guarantees in IP networks for individual application sessions
- Resource reservation: routers maintain state info of allocated resources, QoS req’s
- Admit/deny new call setup requests:

**Question:** can newly arriving flow be admitted with performance guarantees while not violating QoS guarantees made to already admitted flows?
Call Admission

Arriving session must ...

- declare its QOS requirement
  - \textit{R-spec}: defines the QOS being requested
- characterize traffic it will send into network
  - \textit{T-spec}: defines traffic characteristics
- signaling protocol: needed to carry R-spec and T-spec to routers (where reservation is required)
  - \textit{RSVP}
Intserv QoS: Service models
[rfc2211, rfc 2212]

Guaranteed service:
- worst case traffic arrival: leaky-bucket-policered source
- simple (mathematically provable) bound on delay
  [Parekh 1992, Cruz 1988]

Controlled load service:
- "a quality of service closely approximating the QoS that same flow would receive from an unloaded network element."

WFQ

\[ D_{max} = \frac{b}{R} \]
Integrated Services Example

- Install per flow state
Recall RSVP

- Signaling protocol for establishing per flow (soft) state
- Carry resource requests from hosts to routers
- Collect needed information from routers to hosts
- At each hop
  - Consult admission control and policy module
  - Set up admission state or informs the requester of failure
- Decouples routing from reservation
Integrated Services Example: Data Path

- Per-flow classification
Integrated Services Example: Data Path

- Per-flow buffer management
Integrated Services Example

- Per-flow scheduling
How Things Fit Together

Routing Messages

Routing

RSVP

Admission Control

Forwarding Table

Per Flow QoS Table

Route Lookup

Classifier

Scheduler

Data In

Data Out

Data Plane

Control Plane

RSVP messages
QoS in the Internet. Part II.
IETF Differentiated Services

- Want “qualitative” service classes
  - “behaves like a wire”
  - relative service distinction: Platinum, Gold, Silver

- *Scalability*: Simple functions in network core, relatively complex functions at edge routers (or hosts)
  - signaling, maintaining per-flow router state difficult with large number of flows

- Don’t define service classes, provide functional components to build service classes
Diffserv Architecture

**Edge router:**
- per-flow traffic management
- marks packets as in-profile and out-profile

**Core router:**
- per class traffic management
- buffering and scheduling based on marking at edge
- preference given to in-profile packets