Lecture 13 Overview

- Dealing with multiple paths
- Multihoming
- Entact discussion
- Intro to congestion control
Multiple Paths

- Establish multiple paths in advance
  - Good use of bandwidth, withstand failures
- Disseminate link-congestion information
  - Flood thru network, piggyback on data, direct to controller
Adjust Traffic Splitting

- Source router adjusts the traffic
  - Changing traffic fraction, e.g. based on congestion
  - Often use hash-based splitting to prevent packet reordering within a flow
Multi-Homing

- Reliability
  - Reduced fate sharing
  - Survive ISP failure
- Performance
  - Multiple paths
  - Select the best
- Financial
  - Leverage through competition
  - Game 95th-percentile billing model

Stub: an Autonomous System that does not provide transit
Outbound Traffic: Pick a BGP Route

- Easier to control than inbound traffic
  - IP routing is destination based
  - Sender determines where the packets go
- Control only by selecting the next hop
  - Border router can pick the next-hop AS
  - Cannot control selection of the entire path
Inbound Traffic: Influencing Others

- Harder to control than outbound traffic
  - Sender determines where the packets go
- Control only by influencing others’ decisions
  - Static configuration of the providers
  - BGP route attributes sent by the stub
  - Selective advertising of destination prefixes
Inbound Traffic: Multiple Addresses

- Multiple external addresses for a service
  - One IP address for each entry point
- Use DNS to adjust mapping of name to address
  - Give different answers to different clients
  - Adjust over time to performance, cost, traffic, etc.

---

**Provider 1**

- 12.34.1.0/24
- 12.34.1.2

**Provider 2**

- 5.6.7.0/24
- 5.6.7.8
Route Injection

- Minimal impact on current traffic
- Existing approaches are inapplicable

Routing table:

<table>
<thead>
<tr>
<th>prefix</th>
<th>next-hop</th>
<th>AS Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.6.7.0/24</td>
<td>IP2</td>
<td>AS2 AS1</td>
</tr>
<tr>
<td>5.6.7.8/32</td>
<td>IP3</td>
<td>AS3 AS1</td>
</tr>
</tbody>
</table>

5.6.7.8/32 next-hop=IP3

Route injection daemon
CSE 222A – Lecture 13: Traffic Engineering
## Entact Benefits

<table>
<thead>
<tr>
<th>path chosen by Entact</th>
<th>prefixes (%)</th>
<th>wRTT difference (msec)</th>
<th>short-term cost difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>same</td>
<td>88.2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>cheaper &amp; shorter</td>
<td>1.7</td>
<td>-8</td>
<td>-309</td>
</tr>
<tr>
<td>cheaper &amp; longer</td>
<td>5.5</td>
<td>+12</td>
<td>-560</td>
</tr>
<tr>
<td>pricier &amp; shorter</td>
<td>4.6</td>
<td>-15</td>
<td>+42</td>
</tr>
<tr>
<td>pricier &amp; longer</td>
<td>0.1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Congestion Control Overview

- Challenge: how do we efficiently share network resources among billions of hosts?
  - Today: TCP
    - Hosts adjust rate based on packet losses
  - Alternative solutions
    - Fair queuing, RED (router support)
    - Vegas, packet pair (add functionality to TCP)
    - Rate control, credits
Queuing Disciplines

- How to distribute buffers among users/flows
  - When buffer overflows, which packet to drop?

- Simple solution: FIFO
  - First in, first out
  - If packet comes along with no available buffer space, drop it
Fair Queuing

- **Goals:**
  - Allocate resources equally among all users/flows
  - Low delay for interactive users
  - Protection against misbehaving users

- **Approach:** simulate general processor sharing (from OS world)
  - Bitwise round robin
  - Need to compute number of competing flows at each instant
TCP Congestion Problems

- Original TCP sent full window of data

- When links become loaded, queues fill up, leading to:
  - *Congestion collapse*: when round-trip time exceeds retransmit interval this can create a stable condition in which every packet is being retransmitted many times
  - Synchronized behavior: network oscillates between loaded and unloaded
    » Feedback loop
Transport protocols should obey *conservation of packets*
- Use ACKs to clock injection of new packets

Modify retransmission timer to adapt to variations in delay

Infer network bandwidth from packet loss
- Drops $\Rightarrow$ congestion $\Rightarrow$ reduce rate
- No drops $\Rightarrow$ no congestion $\Rightarrow$ increase rate

Limit send rate based on minimum of congestion window and advertised window
Tracking Bottleneck Bandwidth

- Throughput = window size/RTT

- Multiplicative decrease
  - Timeout ➔ dropped packet ➔ cut window size in half

- Additive increase
  - ACK arrives ➔ no drop ➔ increase window size by one packet/window
TCP “Sawtooth”

- Oscillates around bottleneck bandwidth
  - Adjusts to changes in competing traffic

Additive Increase/Multiplicative Decrease

![Graph showing window (in segs) vs. round-trip times](image)

CSE 222A – Lecture 13: Traffic Engineering
Slow Start

- How do we find bottleneck bandwidth?
- Cannot use ACKs to clock without reaching equilibrium
  - Start by sending a single packet
    - Start slow to avoid overwhelming network
  - Multiplicative increase until get packet loss
    - Quickly find bottleneck
    - Cut rate by half
  - Shift into linear increase/multiplicative decrease
Slow Start

- Quickly find the bottleneck bandwidth

![Graph showing Slow Start relationship between window size (in segs) and round-trip times.](image)
Slow Start Problems

- Slow start usually overshoots bottleneck
  - Leading to many lost packets in window
  - Can lose up to half of window size
- Bursty traffic source
  - Will cause bursty losses for other flows
- Short flows
  - Can spend entire time in slow start
    Especially for large bottleneck bandwidth
- Consider repeated connections to the same server
  - E.g., for web connections
For Next Class…

- Read XCP paper

- Project checkpoint due TONIGHT
  - Email 1-2 page summary to me