itmBench: Generalized API for Internet Traffic Managers

Traffic Controllers as Building Blocks
Toward Safely Composing New Network Services

Ibrahim Matta
Computer Science Department
Boston University

Joint work with Gali Diamant, Leonid Veytser, Azer Bestavros, Mina Guirguis, Liang Guo, Yuting Zhang, Sean Chen

CAMAD’04 Workshop, November 29, 2004

http://www.cs.bu.edu/groups/wing
Toward a Road System with Traffic Lights!

**Goal:** more efficient, more predictable
The Internet Traffic Managers (ITM) Architecture

The ITM framework

Identification programs

Adaptive control

Control programs

Bandwidth control
Differentiated control

Traffic characterization
Edge-to-edge control

Capacity planning
Traffic control

Internet domain

Internet domain
TCP Bandwidth Sharing

- TCP needs a slow start period to probe available bandwidth
- Congestion window size depends on when packet loss is detected
- On average, short flows (mice) transfer at a lower rate than long flows (elephants)

TCP Bandwidth Sharing Diagram

- Initial handshake
- Slow-start
- Congestion avoidance

Graph showing TCP sending rate, average sending rate, and linear fit over flow size (pkts)
Elastic TCP-based Tunnels

Providing Soft Bandwidth Guarantees to Aggregates

Elastic tunnel of "m" TCP-friendly flows

Cross traffic

adjust "m"

controller

measured bandwidth

scheduler

TCP

monitor

IP
eTCP Controllers

Naïve control: \[ m(t+1) = \frac{B^*}{b(t)} m(t) \]

P control: \[ m(t+1) = K_p e(t),\quad e(t) = B^* - b(t) \]

PI control: \[ m(t+1) = K_p e(t) + \tau_i K_i e(t) \]
Simulations

Naïve vs PI vs No Control

---

Naïve:  - big overshoot
        - aggressive reaction
        - not good for highly dynamic environments

PI:    + more steady
        + less overshoot
        + less jitter

---

Bandwidth achieved

# active ITM-TCP connections
SaTS: Size-aware TCP Service

**Preferential Early Dropping**

Proactively send congestion notification even before queue is full

- **queue length** $q$
- **dropping probability** $p_i$

<table>
<thead>
<tr>
<th>High priority</th>
<th>Low priority</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Size-threshold Controller

Optimization constraint: Penalty < 3%

NLANR Trace, Indiana University
itmBench: Generalized API for ITMs

- **itm_mod** serves as intermediate layer between applications and TCP/IP stack.
- Other modules register with it to receive service.
- Keeps record of registered applications and transfers packets according to given specifications.
- Currently uses netfilter to interact with stack.

Diagram:

```
application → libitm

registration → packets
itm_mod

SaTS_mod → eTCP_mod

input

itm_queue

TCP/IP

kernel

user space
```
Registering with itm_mod

Kernel space Applications

- Kernel modules register directly with itm_mod
- Each module provides a specific service or implements an application
- Packets can be dropped, changed or left untouched then re-injected to stack

User space applications

- Register with itm_mod, using libitm
- itm_queue communicates with itm_mod to receive packets like a regular kernel application
- itm_queue demultiplexes packets to applications
An Event-driven API

- A registering application needs to specify functions to handle event:

  - Classify
  - Monitor
  - Process
  - Control
Synchronous Events

- Use `itm_register_hook` to register with `itm_mod`, specifying pointers to functions to be activated on event.
Asynchronous Events

- Use `itm_register_timer` to register with `itm_mod`, specifying pointers to functions to be activated when timer expires.

- Event is timer-triggered: specify amount of time to wait before event occurs. After handling, can re-register to trigger again at a later time.
eTCP using itmBench API

Packet arriving at source ITM

Classify
- Incoming eTCP ACK
- Registered client

Periodic Control
- adjust eTCP tunnel’s strength

Monitor
- update eTCP tunnel’s goodput ...
- update eTCP tunnel’s throughput ...

Process
- forward back to client
- encapsulate & send over eTCP tunnel
SaTS using itmBench API

Packet arriving at source ITM

Classify

Registered client

Monitor

update flow’s size ...

Process

mark class in packet’s header

Periodic Control

adjust size threshold
Food for Thought:
Safe Composition

- Is it “safe” for an itmBench programmer (e.g. network manager or service provider) to compose API “functions” or even “services”?
  - What if a PI controller is replaced by a PID controller?
  - What if multiple bandwidth tunnels are cascaded?
  - What if a SaTS service is cascaded with an eTCP tunnel?
Questions?

For more information, please check
http://www.cs.bu.edu/groups/itm

For more WING projects, please check
http://www.cs.bu.edu/groups/wing