XCP Deployment Challenges

Aaron Falk, Eric Coe, Ted Faber, Jasmeet Bagga
USC Information Sciences Institute
Outline

- Some Background
- Network Processor Implementation
- XCP Sharing with Reno flows
- XCP and Non-XCP Queues
- XCP in Space
- Other project notes
Background
XCP in a Nutshell

- XCP is a congestion control protocol published in 2003 by Katabi, et al.
- End-systems use explicit signaling to tell routers their preferred send rate.
- Routers...
  - make a per-flow allocation (without keeping per-flow state),
  - inspect incoming packets, and
  - reduce the throughput request to match the allocation (if necessary).
ISI’s XCP Development

- Our objective: Take XCP from theory to reality
- To get there:
  - Build & test a kernel implementation ✓
  - Evaluate the cooler aspects:
    - Rapid convergence ✓
    - Good performance over large BDP, RTT ✓
  - Write a protocol specification ✓
  - Develop deployment strategies (today)
Deployment Questions about XCP

- How hard does an XCP router need to work?
- Can XCP share capacity fairly with Reno TCP?
- How well can XCP perform when some queues along the path are not XCP-capable?
- What benefits can XCP provide when incrementally deployed?
Today’s talk will be a snapshot of where we are in addressing these questions.
XCP Network Processor Implementation

Work by Eric Coe & Yuri Pryadkin
XCP Implementation on Intel’s IXP2400 Network Processor

- Motivation
  - Show that the XCP algorithm can be implemented in hardware
- Benefits of this Platform
  - Intel’s IXP2400 supports OC-48 line rates
  - Feature set more closely resembles IP routers
- Issues Being Addressed
  - Eliminating division in the fast path
  - Eliminating floating point calculations
  - Fixed-point scaling of header variables for throughput & RTT ranges
XCP on IXP2400

High Speed Data Path

μengineThread → IP Routing Decision → Packet Arrival → Packet Departure → Transmit Packet

μengineThread
Receive Packet

μengineThread
Estimate Control Timer

μengineThread
Queue Measurement Timer

Intel XScale

Control Path

A. Falk - USC/ISI © 2005
XCP & Reno Sharing

Work by Jasmeet Bagga & Ted Faber
XCP is More Sensitive Than Reno

- If XCP and Reno share a link & buffer, XCP will back off as Reno fills the queue
- Need to *fairly* partition capacity for XCP
- Our definition of fairness: 
  *The average XCP flow throughput should be about equal to the average Reno flow throughput*
Fair Sharing Between Reno and XCP

- Katabi suggests two schemes:
  - Core Stateless Fair Queuing
    - Requires per flow state at the edge routers – \textit{would like to avoid}
  - TFRC-like approach
- TCP Throughput Equation is used to manipulate the Reno drop rate to achieve comparable Reno & XCP average flow throughputs
- Requires no per flow state – but we are concerned about stability, and accuracy of this approach
Another Approach

• If we knew the number of Reno flows, we could estimate average per-flow throughput…

• Evaluating hashing function to obtain an estimate of flow count

• Next question: Can we do it at line rate?
XCP & Non-XCP Queues

Work by Ted Faber
Not All Queues Will Be XCP Capable

- An XCP sender relies on feedback from bottlenecks to find the correct throughput
- What happens if the bottleneck isn’t XCP capable?
  - A: switch to end-to-end congestion control, e.g., Reno
  - (Aside: if the non-XCP queue is not a bottleneck, don’t worry about it)
- Challenge: identifying when the bottleneck moves to & from a non-XCP queue
A Strategy for Adapting to Congestion in Non-XCP Queues

- ECN or 3 DUPACKs indicates a non-XCP bottleneck
- Sender switches to end-to-end congestion control
- Sender continues to advertise Throughput, Delta_Throughput and monitor Feedback
- When Feedback becomes negative, bottleneck has moved to an XCP queue
- Need to be aware of when XCP routers are ‘shuffling’ vs. responding to congestion – new flag in header
Plenty of Questions

- How well does this work? Need to implement and simulate
- Are we using the right algorithms to switch between regimes?
- Does delaying the switch have adverse effects on the network?
XCP & Partial Deployment

Work by Aman Kapoor & Aaron Falk
XCP in Space

• Future satellite networks are a good fit for XCP deployment
• Very large BDP: RTTs: ~sec; link rates: ~Gbps
  • Congestion avoidance ‘recovery time’ for 500ms RTT, 1Gbps flow would be 5.8 hours:
  • Reno performs poorly
• Dynamic bandwidth management (BoD, DAMA) may be used to allocate link access
  • RTT may vary significantly
  • Vegas likely a perform poorly
• XCP should perform well in this environment
Performance-Enhancing Proxies (PEPs, RFC3135) split TCP connections and run another protocol between.

- ISI has developed and is testing an XCP PEP.
- Data is sent much faster than end-to-end Reno.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Average</th>
<th>Std. Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reno e2e</td>
<td>123.38</td>
<td>1.28</td>
</tr>
<tr>
<td>XCP e2e</td>
<td>1.47</td>
<td>0.32</td>
</tr>
<tr>
<td>XCP PEP</td>
<td>1.61</td>
<td>0.26</td>
</tr>
</tbody>
</table>
Proxy Next Steps

- Evaluate XCP performance over intermittent links and links with ARQ
- Consider performance with more diverse traffic
  - Trial deployment?
- Extend proxy to enable XCP for traffic engineering – another partial deployment scenario
- See http://nms.lcs.mit.edu/~dina/texcp.html
Other Project Notes

- XCP available in upcoming ns-2 (v2.28) release
- draft-falk-xcp-spec-00.txt
People

- Aaron Falk, ISI
- Ted Faber, ISI
- Bob Braden, ISI
- Yuri Pryadkin, ISI
- Padma Haldar, ISI
- Eric Coe, student
- Aman Kapoor, student
- Jasmeet Bagga, student
- Nirav Jasapara, student
- Dina Katabi, MIT
- John Wroclawski, MIT
XCP Project Info

- http://www.isi.edu/isi-xcp
- source code for XCP end-system, router, PEP
- draft specification
- mailing list information
This material is based upon work supported by the National Science Foundation under Grant No. ANI-0334186.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF) and USC/Information Sciences Institute.
The End