ETH zürich

Enabling Internet-Wide Deployment of Explicit Congestion Notification

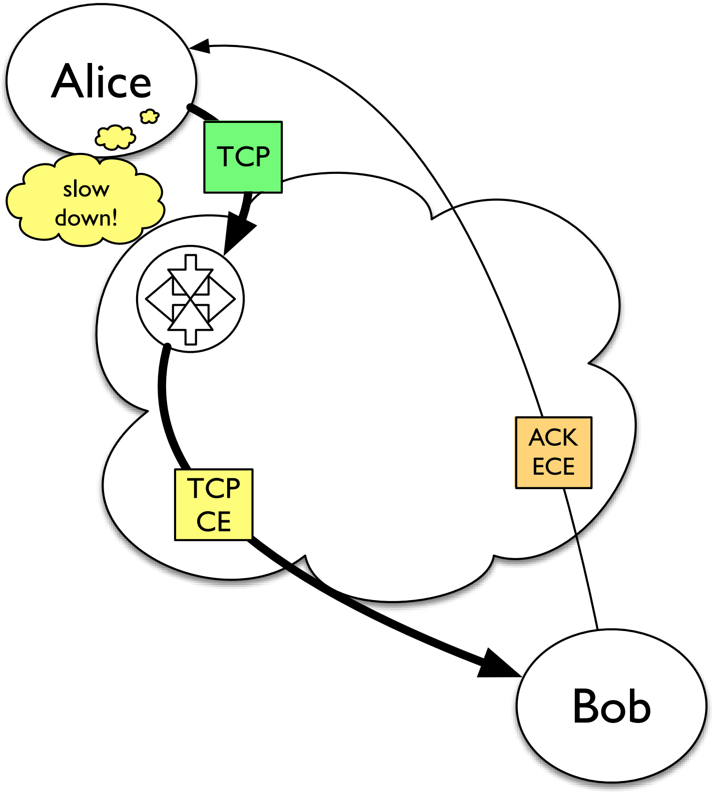
Brian Trammell, Mirja Kühlewind, Damiano Boppart,
Iain Learmonth, Gorry Fairhurst, and Richard Scheffenegger

Internet Congestion Control Research Group
Dallas, Texas, IETF 92, 23 March 2015

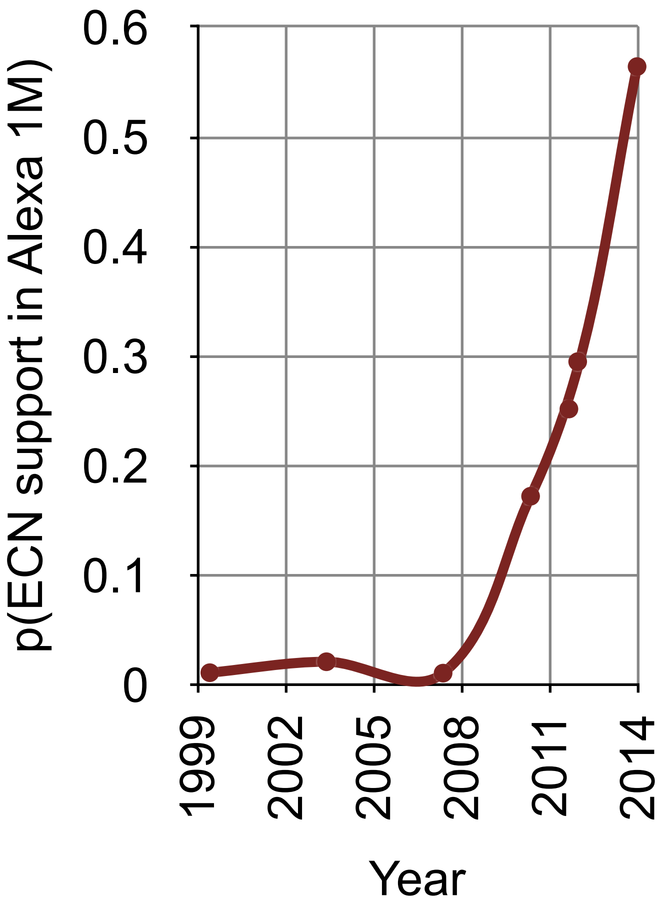
mPlane

The Problem

- Explicit Congestion Notification (ECN) defined in RFC 3168
  - 15 years ago!
- Idea: routers mark packets to signal congestion
- Deployment largely failed
  - Rebooting routers, broken middleboxes, overprovisioning
- ECN is relevant again
  - Changing network environment, changing requirements for ECN (e.g. DCTCP).



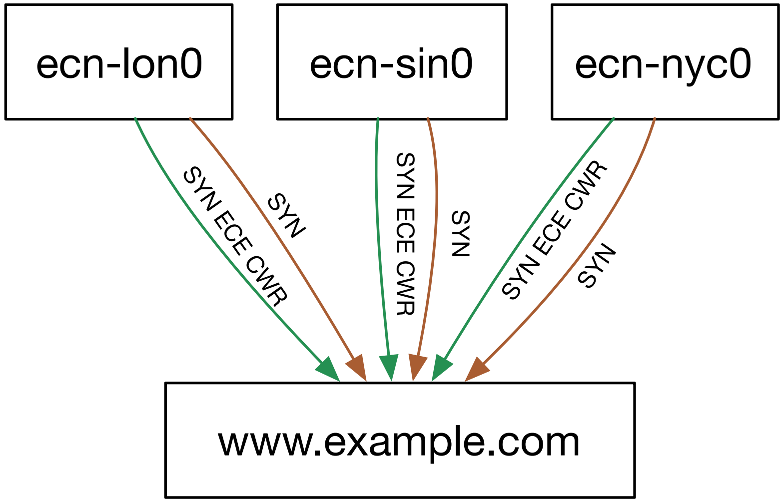
In the meantime…



- ECN negotiation (for TCP) uses additional flags in the handshake
  - SYN ECE CWR
  - SYN ACK ECE
  - ACK
- Linux defaults to passive ECN negotiation (i.e., server will negotiate ECN if asked)
  - increasing server deployment
  - but no client usage (PAM 2013)
- Question: **can we leverage client side defaults to drive deployment of ECN?**

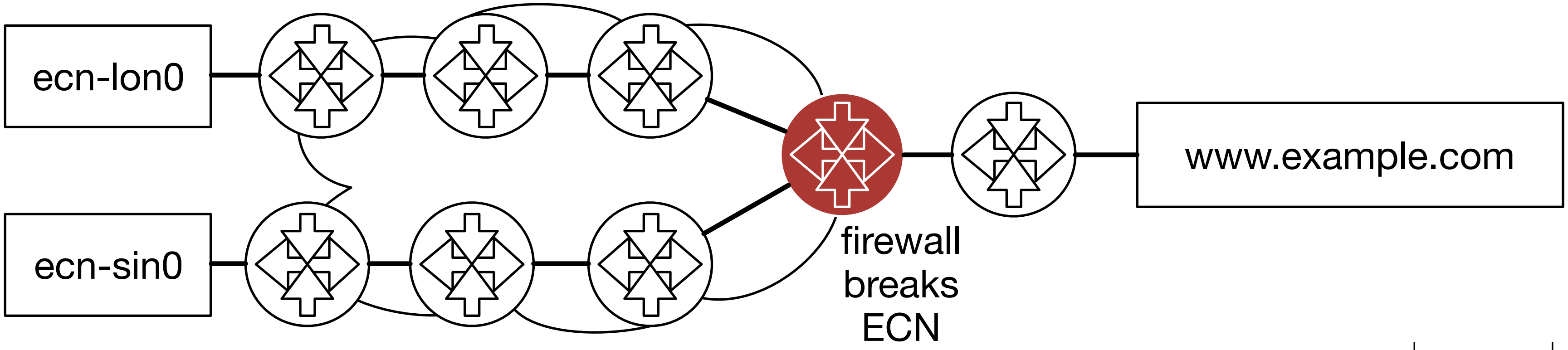
Connectivity risk of client-side ECN default

- **Methodology**: run *n* trials from *m* vantage points, comparing connectivity with ECN negotiation enabled to that with ECN negotiation disabled, using the Linux `tcp_ecn` sysctl.
  - Always succeeds, regardless of ECN → OK
  - Always fails, regardless of ECN → simply broken
  - Always succeeds without ECN, always fails with ECN → *ECN-dependent connectivity*
  - ECN dependent connectivity from only some vantage points → *path-dependent ECN-dependent connectivity*
- Target the top million Alexa webservers from three vantage points from digitalocean.com



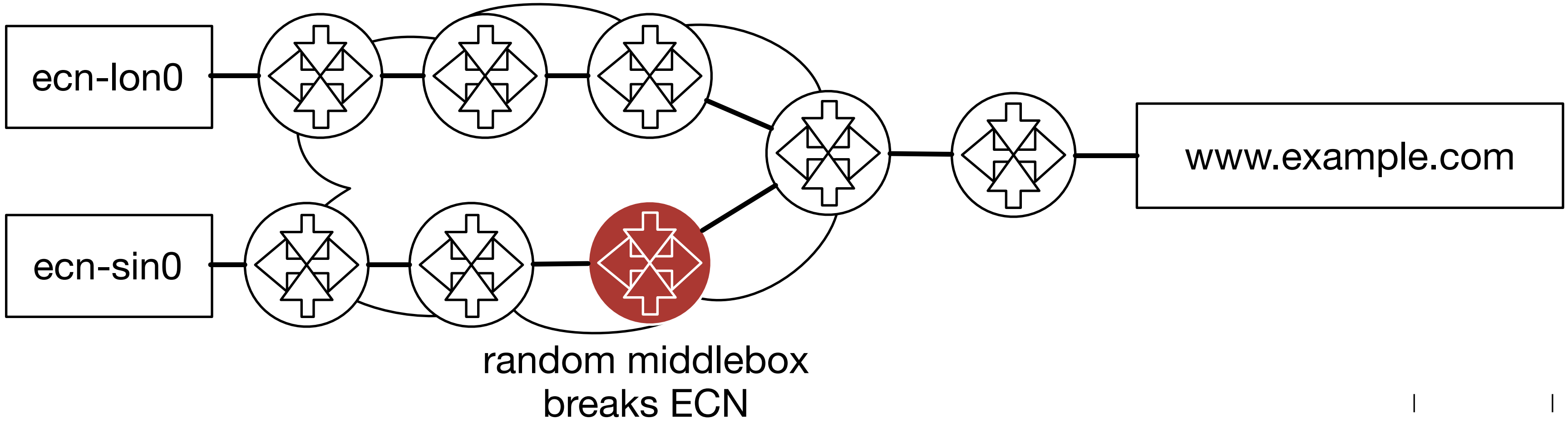
Endpoint-dependent connectivity dependency

- If the box breaking ECN is close to the server, fallback as in RFC 3168 can save us:
  - retransmitted SYN ECE CWR is SYN only, no ECN.
- ~0.4% of the paths, risk of increased connection latency.
  - much less than ~0.4% of the traffic
- Probably a firewall → content provider or CDN can fix this problem with relatively little effort in an ECN-by-default world.



Path-dependent connectivity dependency

- This is worse news: ECN breaks on the path outside the content provider's network.
  - Content provider can't easily fix the problem
  - Rerouting might cause ECN to break mid-flow
- Definitely seen about on about 2.5 per 100'000 hosts…
  - …and a third of these are GoDaddy parking sites
  - …we tried to use traceroute to find the rest, but it lied to us



Connectivity Dependency Results

Table 1. Connectivity statistics, of 581,737 IPv4 hosts and 17,029 IPv6 hosts, all vantage points, 27 Aug - 9 Sep 2014

IPv4		IPv6		
hosts	pct	hosts	pct	description
553805	95.20%	14889	87.43%	Always connected from all vantage points
3998	0.69%	1594	9.36%	Never connected from any vantage point
8631	1.48%	138	0.81%	Single transient connection failure
11999	2.06%	324	1.90%	Non-ECN-related transient connectivity
578433	**99.43%**	**16945**	**99.50%**	**Total ECN-independent connectivity**
2193	0.38%	13	0.08%	Stable ECN dependency near host
15	0.00%	0	0.00%	Stable ECN dependency on path
34	0.01%	3	0.02%	Potential ECN dependency on path
201	0.03%	0	0.00%	Temporal ECN dependency
2443	**0.42%**	**16**	**0.09%**	**Total apparent ECN-dependent connectivity**
862	0.15%	69	0.41%	Inconclusive transient connectivity

Connectivity Depends on OS and Rank

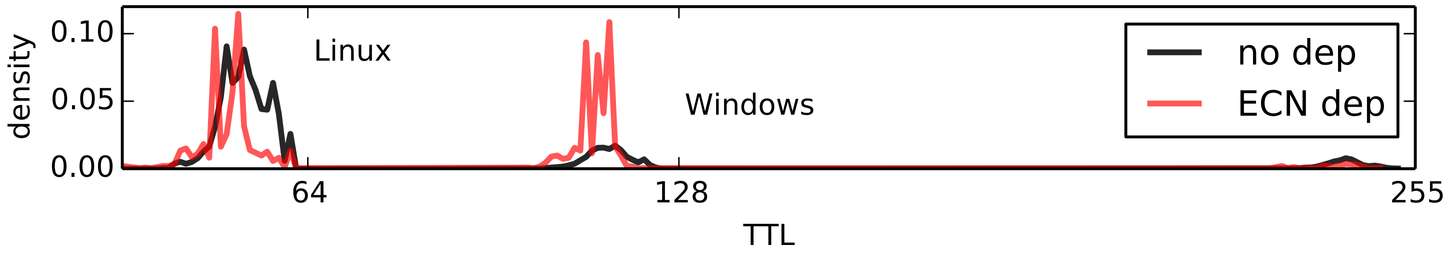


Fig. 1. TTL spectrum of ECN-dependent and -independent connectivity cases

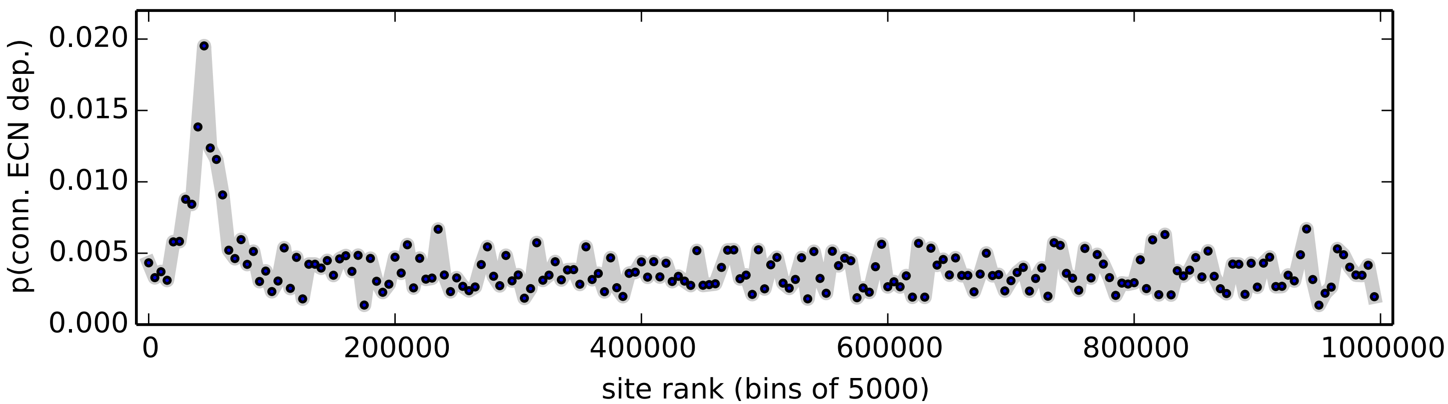


Fig. 2. Proportion of sites failing to connect when ECN negotiation is requested

ECN Negotiation Results

Table 2. ECN negotiation statistics, of 581,711 IPv4 hosts and 17,028 IPv6 hosts, all vantage points, 27 Aug - 9 Sep 2014, compared to previous measurements.

IPv4		IPv6		2011	2012	
hosts	pct	hosts	pct	pct[5]	pct[2]	Description
326743	**56.17%**	**11138**	**65.41%**	11.2%	29.48%	**Capable of negotiating ECN**
324607	55.80%	11121	65.31%	–	–	...and always negotiate
2136	0.37%	17	0.11%	–	–	...sometimes negotiate, of which...
107	0.02%	1	0.01%	–	–	negotiation depends on path
27	0.02%	0	0.00%	–	–	sometimes reflect SYN ACK flags
248791	43.23%	3961	26.23%	82.8%	70.52%	Not capable of negotiating ECN
2013	0.35%	83	0.48%	–	–	...and reflect SYN ACK flags
6177	1.06%	1929	11.33%	–	–	Never connect with ECN (see §3.1)

The trend of increasing willingness to negotiate ECN continues…

ECN signaling results

Table 3. Relationship between ECN IP and TCP flags (*expected cases in italics*)

	IPv4 (N=581711)			IPv6 (N=17028)		
Marking	ECN	Reflect	No ECN	ECN	Reflect	No ECN
only ECT(0)	*315605*	693	1995	*8998*	1	46
ECT(0) + ECT(1)	0	0	0	4	1	7
ECT(0) on SYN ACK	7780	0	46	89	0	82
only ECT(1)	3	1	17	0	10	12
ECT(1) on SYN ACK	4	0	16	7	0	31
only CE	11	1	7	0	0	48
CE + ECT	5	2	0	23	66	39
CE on SYN ACK	11	0	5	22	0	87
none	6939	1343	*243150*	2013	5	*3694*

…but signaling is less reliable, and the situation is worse on IPv6 than IPv4.

(And of ~5 million flows, we saw only two legitimate CE markings.)

Conclusions and future work

- Can we safely leverage client-side defaults to drive ECN deployment?
  - ***Yes.***
- What is the risk to connectivity (to popular websited) of doing so?
  - *< **$O(10^{-4})$** on a path basis when fallback as in RFC 3168* is used.*
  - *≪ **$O(10^{-4})$** weighted by traffic volume (how much less depends on the model)*

- Once ECN is negotiated, signaling anomalies in ~2% of cases may interfere.
  - ∴ the next step to making the world safe for ECN is defining methods for detecting and reacting to signaling failures in the transport stack.

- What we're doing next:
  - defining these signaling fallback methods (IETF)
  - measuring the situation for non-web services and access networks
  - making continuous measurement available at http://ecn.ethz.ch

*Apple and Microsoft do this already; we have a patch for the Linux kernel