Flow label for equal cost multipath routing in tunnels

draft-carpenter-flow-ecmp-01

Brian Carpenter
University of Auckland

March 2010

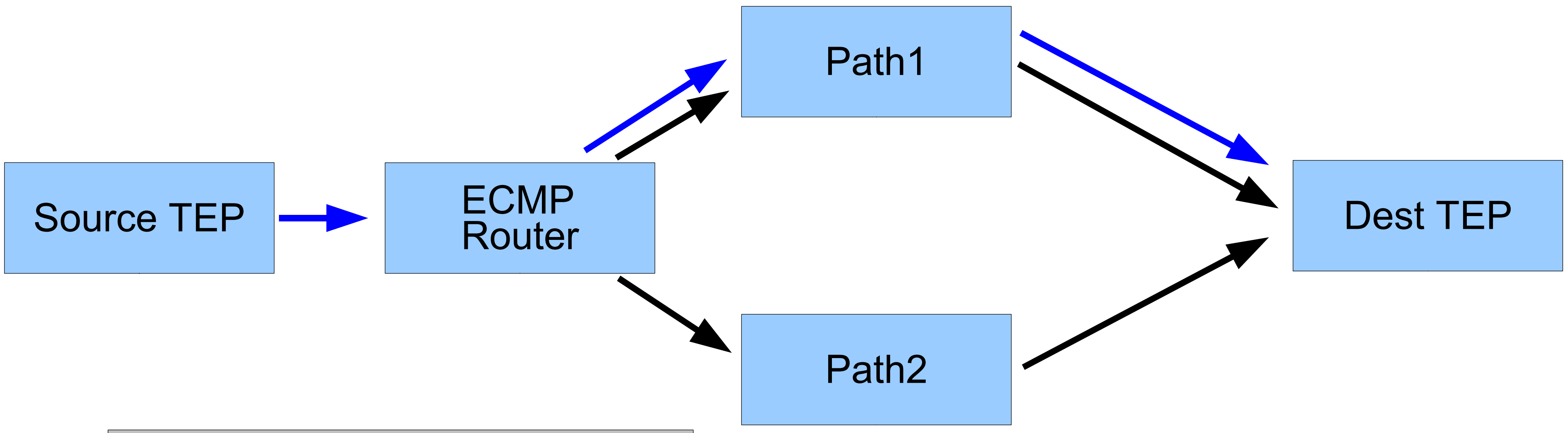
ECMP goals

- Roughly equal share of traffic on each path.
- Work-conserving method (no idle time when queue is non-empty).
- Minimize or avoid out-of-order delivery for individual traffic flows.

Basic approach to ECMP

- If there are N equally good paths to choose from, then form a hash code modulo(N) from each packet header
- Use the resulting value to select a particular path.
- Typically, hash the 5-tuple {dest addr, source addr, protocol, dest port, source port}.

The problem with tunnels



Source TEP

ECMP Router

Path1

Path2

Dest TEP

Normal traffic split by ECMP.
Tunnel traffic all has same 5-tuple; no split.

Proposed solution

- For foo-in-IPv6 tunnels, the TEP sets a flow label per user flow in the *outer* packet
  - For IP-in-IPv6, the flow label is based on the 5-tuple of the *inner* packet
  - It should be well distributed (pseudo-random)
- The ECMP router hashes a 6-tuple, the normal 5-tuple plus the flow label
  - works the same as before for non-tunnel traffic (and even better if flow label is set)
  - also splits tunnel traffic
  - fully conformant with RFC 3697

Update to the IPv6 flow label specification

draft-carpenter-6man-flow-update-00

Brian Carpenter
University of Auckland

Sheng Jiang
Huawei

March 2010

Why?

- RFC 3697 says:
  - Flow label must not be changed en route.
  - Nodes must not assume any mathematical or other properties of Flow Label values
  - Router performance should not depend on the distribution of Flow Label values... Flow Label bits *alone* make poor material for a hash key.
- These rules have caused difficulty for almost all proposed use cases.

What the use cases tell us

- Type 1: QoS or routing proposals.
  - These want to encode QoS or routing semantics in the flow label, and often want this done by the ingress router not the source.
  - (A bit like diffserv on steroids, or intserv on slimming pills; or MPLS-like.)
  - Definitely break the rules in RFC 3697.
  - There are quite a few such proposals around.
- Type 2: Pseudo-random based proposals
  - Such as draft-blake-ipv6-flow-label-nonce and draft-carpenter-flow-ecmp
  - Rely on that subtle “*alone*” in RFC 3697

Proposal (1)

- Update RFC 3697
- Use the MSB of the flow label to separate Type 1 and Type 2 use cases
- Knowing that non-zero flow labels are vanishingly rare today, we can devise rules that should avoid any backwards compatibility issues.

Proposal (2)

- Flow Label ≠ 0 and MSB = 0
  - Flow label follows all rules of RFC 3697 (as far as the remaining 19 bits go)
- Flow label ≠ 0 and MSB = 1
  - Locally defined usage applies, RFC 3697 does not apply.
  - Clear remaining 19 bits before exporting packet from local domain
- Flow label = 0
  - Locally defined usage allowed, but label must be set back to 0 before delivering or exporting packet
    - this will need a flag bit in the local usage

Consequences (1)

Considering packets sourced within local domain:

- Hosts wanting RFC 3697 behavior set flow labels between 1 and 0x7FFF
- Hosts wanting local behavior set flow labels between 0x80000 and 0xFFFFF
- Hosts that set zero flow labels are unaffected
  - their traffic might benefit from local behavior
  - but the label is delivered as zero
- Receiving hosts that ignore the flow label are unaffected.
  - updated hosts *may* interpret the MSB

Consequences (2)

Considering packets entering or leaving local domain:

- Incoming packets
  - if MSB=0, RFC 3697 applies
    - if flow label = 0, allow local behavior?
  - if MSB=1, may benefit from local behavior.
- Outgoing packets
  - if MSB=0, RFC 3697 applies (preserve label)
  - if MSB=1, may benefit from local behavior in other domains
    - clear the other 19 bits? or clear the whole label?

Note that this is not exactly what the 01 draft says.

Alternative approach

- Do not use MSB as flag.
- Define a special DSCP meaning “locally defined flow label semantics in use”
- Use this instead of the MSB in the previous rules.
- Issues
  - DSCP values themselves are locally defined according to RFC 2474: no universal values.
  - Mixes diffserv and flow label semantics

Discussion

1. Is the basic idea useful?
2. Is the DSCP alternative better?
3. Detailed rules for domain boundary?