Multipath bonding at Layer 3

Maciej Bednarek (ETH Zurich), Mirja Kühlewind (ETH Zurich), Guillermo Barrenetxea Kobas (Swisscom), Brian Trammell (ETH Zurich)

July 16, 2016 - Applied Network Research Workshop (ANRW)
Overview

- **Motivation**
  Operator’s demand for aggregation of DSL and mobile capacity

- **Layer 3 Bonding Solution**
  Architecture and Scheduling Algorithm

- **Implementation**
  Packet mangling, scheduling, and re-ordering

- **Evaluation**
  Single Flow and TCP cross traffic

- **Conclusion**
  Works but further work needed…!
Motivation: Aggregation of DSL and mobile capacity

- DSL capacity is not sufficient to e.g. serve HD video service
- MPTCP proxy only suitable for TCP traffic
**Bonding Architecture:**
Costumer and Provider Bondings Gateways

- **Ingress:** accepts traffic, schedules transmission & adds SEQ#
- **Egress:** takes traffic from bonding interface, re-orders & strips SEQ#, sends loss report to ingress
Scheduling Algorithm: Adaptive Weight Increment (AWI)

**Goal:** fill fixed link first, use mobile link for excess traffic demand only

**AWI using Weighted Round Robin (WRR)**

- fixed weight for fixed line: \( w_{\text{fixed}} = 50 \)
- dynamic calculation for mobile line (initially \( w_{\text{mobile}} = 0 \)):

\[
 w_{\text{mobile}} \leftarrow k \times \frac{\text{pkt}_{\text{lost}}}{\text{pkt}_{\text{sent}}} \times w_{\text{fixed}} 
\]

control parameter
Scheduling Algorithm: Initial Weight Increment (IWI)

**Goal:** react quickly when congestion is arising

If $w_{mobile} = 0$ & loss is reported:

- increases $w_{mobile}$ by the number of lost packets

Note: $w_{mobile}$ is clamped to a maximum value $w_{mobilemax} = 50$
Scheduling Algorithm: Delayed Weight Decrement (DWD)

**Goal:** shift load back to the fixed line without inducing loss by shifting the load too quickly

If no loss reported for $T_{dwd}$:

- decrement $w_{mobile}$ by one for each interval $T_{report} = 50ms$

Note: We investigate different values for $T_{dwd}$ but it must be a multiple of $T_{report}$ (as loss reports are only received every $T_{report}$ milliseconds)
Implementation: Bonding Ingress

intercepts packets using Netfilter queues (in `OUTPUT` chain) and forward to userspace

- **Packet Mangling**
  - Control packets from the egress (loss reports) will be discarded
  - Data packets: sequence number added & forwarded for scheduling
    - Generic Routing Encapsulation (GRE) Sequence Number and Key fields could be used

- **Scheduling**
  - Decides about netfilter mark (`fwmark`) to map data packet to the right output queue using `iptables`
  - Counts the number of packets sent on each interface (`pkt_sent`)
Implementation: Bonding Egress

intercepts all incoming UDP packets using Netfilter queues (in PREROUTING chain)

• Re-ordering
  1. New packet received:
     • **forward** packet directly if $SEQ# = last_{accepted} + 1$ (or the first of a new flow) and update last_{accepted}
     • **enqueue** packet if $SEQ# > last_{accepted} + 1$ (and remember timestamp)
     • **discard** packet if $SEQ# < last_{accepted} + 1$ (as it has been assumed to be lost)
  2. Further check other packets in queue (and update last_{accepted}):
     • **forward** first packet in queue if now $last_{accepted} + 1 = SEQ#$ of queued one
     • **forward** also if now - $T_{dwd} > timestamp$ (missing packet is assumed to be lost)
Evaluation: Experimental setup

- Two Linux Debian Wheezy machines (client & server)
- 1492 bytes UDP packets (28 bytes UDP/IPv4 header, 4 bytes for SEQ#, and 1460 bytes of dummy payload)
- TCP cross traffic: file transfer from a public server (cdimage.debian.org) with 50ms to client
- DSL link is shaped to a maximum rate of 64 Mb/s and stable 13ms delay (measured)
- Swisscom’s Huawei E3276s LTE stick with about 60Mb/s (and variable delay of 25 - 45ms)
Evaluation:
Results for a single flow

$k=0.1$

- Loss rate 6.25%
- $T_{dwd} = 50\text{ms}$

- Loss rate 4.72%
- $T_{dwd} = 250\text{ms}$

- Loss rate 4.1%
- $T_{dwd} = 500\text{ms}$

$k=0.5$

- Loss rate 1.79%
- $T_{dwd} = 50\text{ms}$

- Loss rate 0.53%
- $T_{dwd} = 250\text{ms}$

- Loss rate 0.34%
- $T_{dwd} = 500\text{ms}$

$k=1$

- Loss rate 1.14%
- $T_{dwd} = 50\text{ms}$

- Loss rate 0.29%
- $T_{dwd} = 250\text{ms}$

- Loss rate 0.15%
- $T_{dwd} = 500\text{ms}$

$k$ and $T_{dwd}$ provide trade-off between aggressiveness and responsibility
Evaluation: Results with TCP cross traffic

- $T_{dwd} = 50\text{ms}$: TCP flow only gets spare capacity
- $T_{dwd} = 1000\text{ms}$: UDP traffic permanently shifted to mobile link

Operator can decide how TCP-friendly the algorithm should be
Conclusion

- **Goal**: Aggregation of DSL and mobile capacity for excess traffic
- Layer 3 bonding solution
  - Ingress: Packet mangling (SEQ#) and scheduling that adapts $w_{\text{mobile}}$ dynamically
  - Egress: Re-ordering buffer
- Evaluation of parameters $k$ and $T_{dwd}$
  - Trade-off between aggressiveness and responsibility
- **Future Work**
  - Interoperation with presently deployed MPTCP proxies
  - Middlebox cooperation to indicate if re-ordering sensitivity
Evaluation

Results for a multiple UDP flows