PEPsal: a Performance Enhancing Proxy for TCP satellite connections (and future research directions at UoB)

Carlo Caini, Rosario Firrincieli, Daniele Lacamera

University of Bologna Italy

ccaini@arces.unibo.it, rfirrincieli@arces.unibo.it, root@danielinux.ne

Outline

- PEPsal
  - PEPsal architecture
  - PEPsal implementation
  - Performance
  - Service
- Future research directions
  - DTN

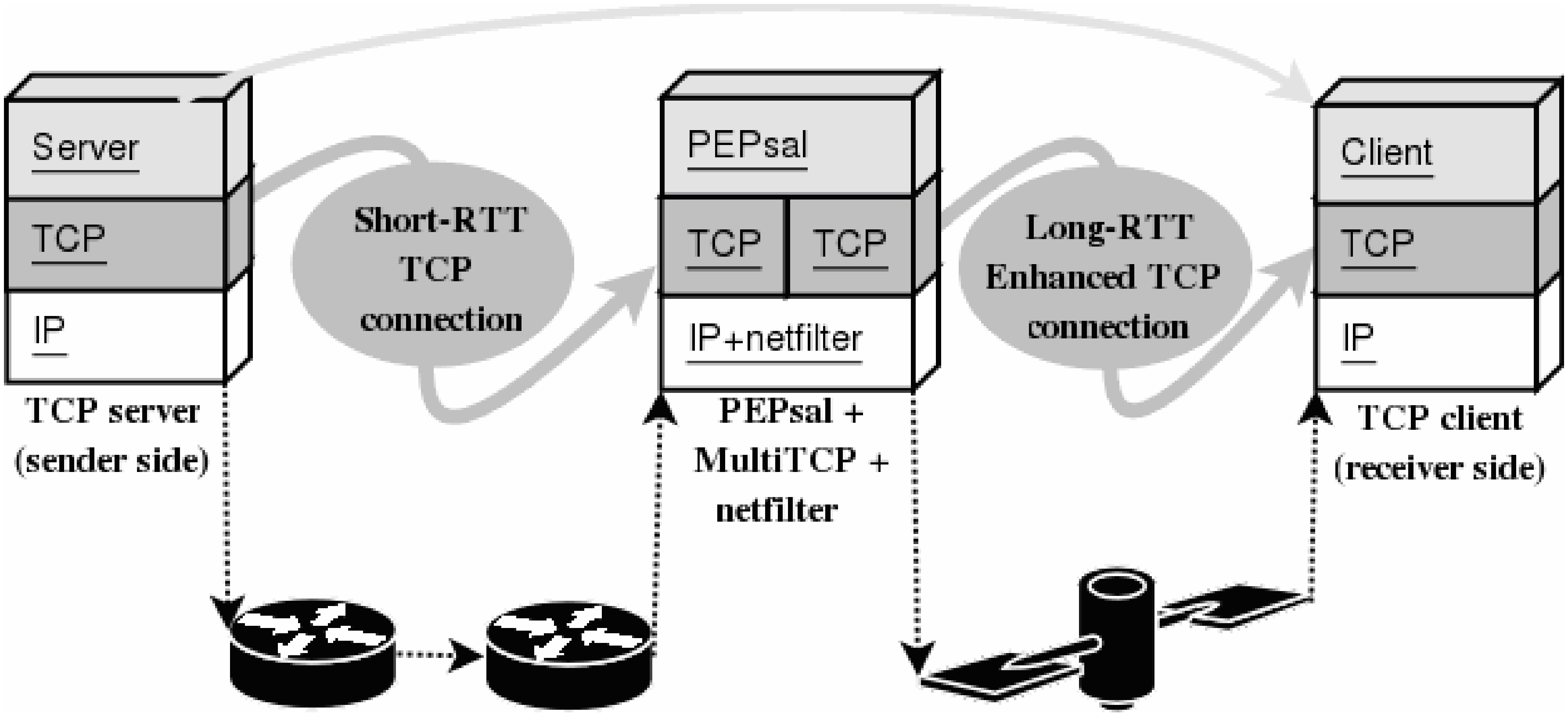
Introduction

- TCP satellite connections strongly penalized by
  - long RTTs (about 600 ms for GEO)
  - possible presence of random losses on the sat channels
- PEPs' rationale
  - To isolate satellite channel impairments
- PEPsal in a nutshell
  - "TCP-splitting" concept
    - Standard TCP on the terrestrial legs
    - TCP Hybla on the satellite leg (but all Linux TCP variants allowed)
  - Adopted with success by a US satellite provider
  - Free Software running on Linux (sf.net/projects/pepsal)

PEPsal Classification

- Following RFC 3155
  - Multilayer
    - Application, Transport, IP
  - Integrated
    - Single-box on the sat gateway
  - Asymmetric
    - Symmetric possible with receiver side modifications
  - Transparent

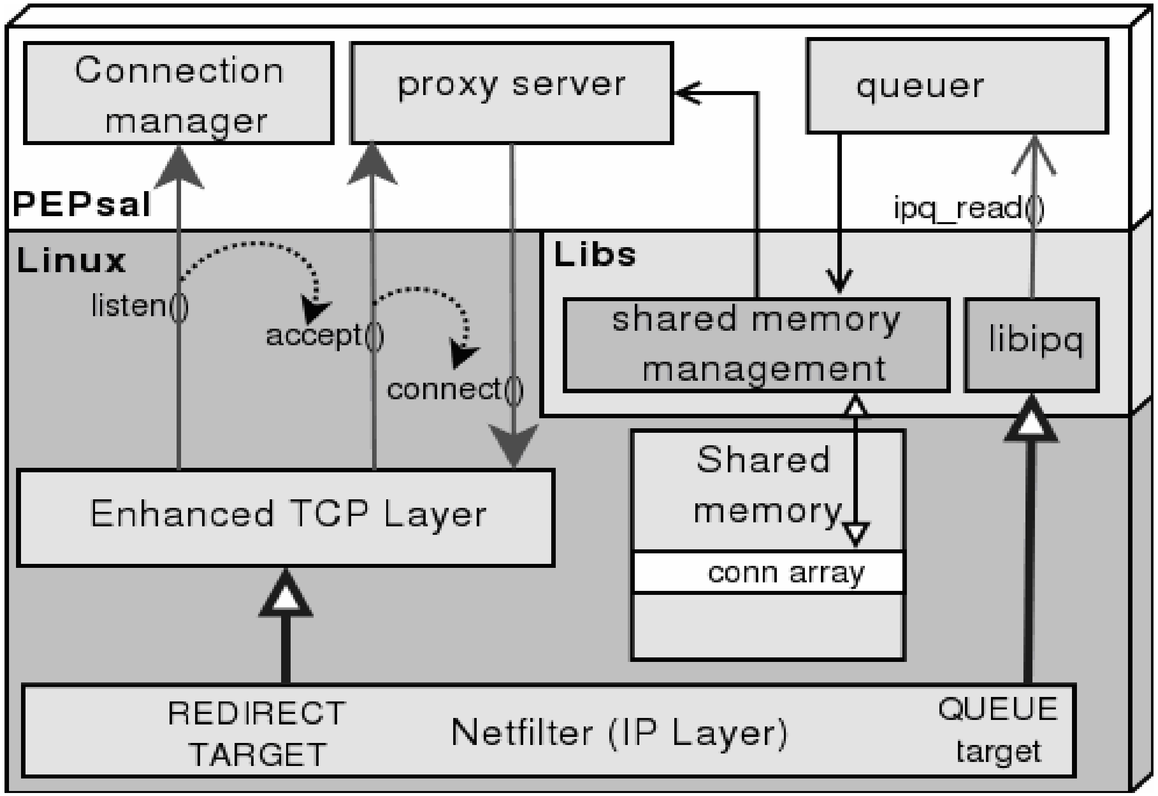
PEPsal Architecture



The UoB PEPsal implementation

Multi-threaded, based on Linux 2.6 + netfilter + libnfqueue

- At first developed for academic purposes, then used with TCP Hybla on the sat segment to provide real one-way satellite Internet access



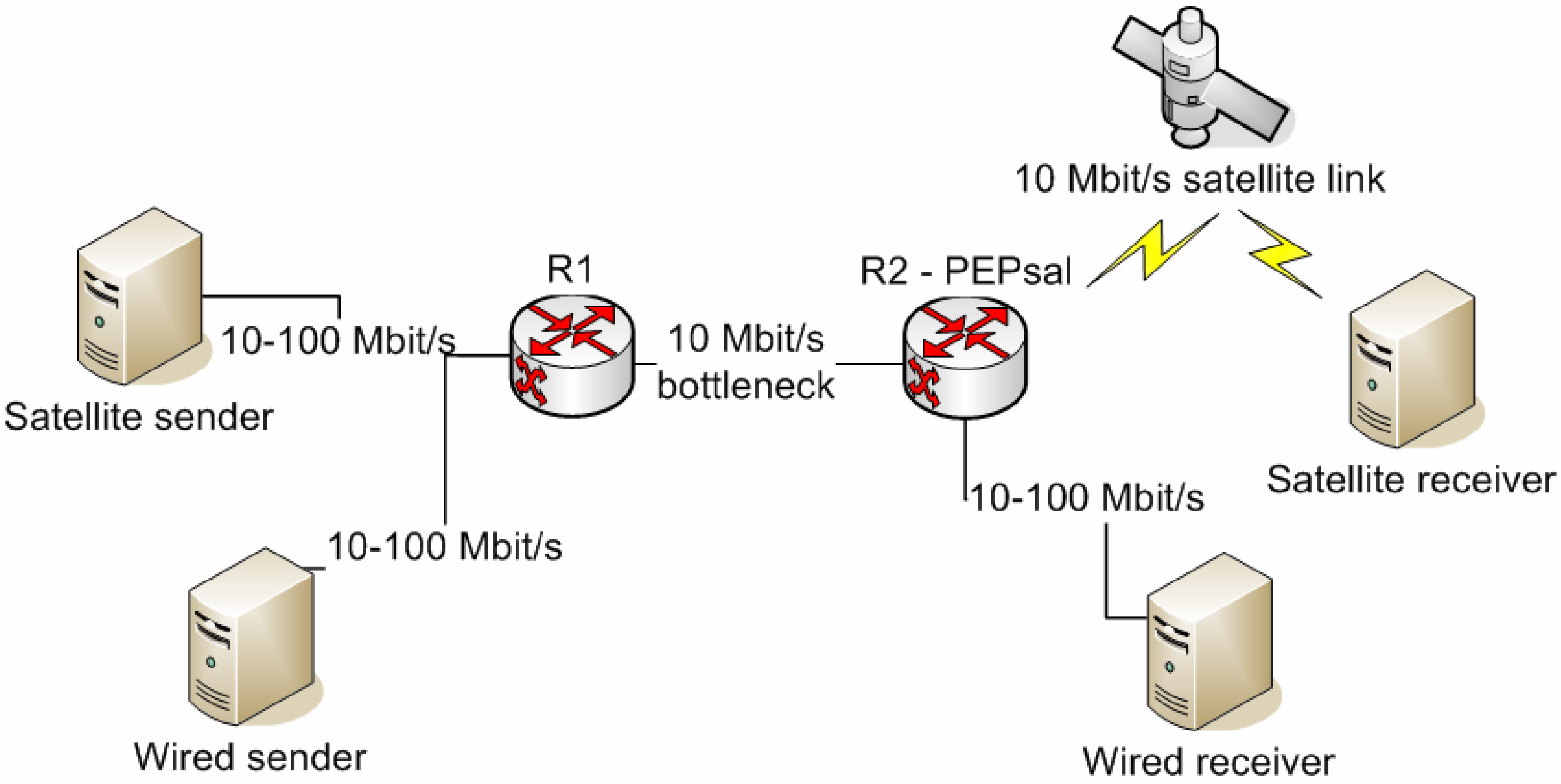
Where does performance improvement come from ?

- TCP-splitting: end-to-end satellite connection split in
  - a “terrestrial” connection (short RTT, error free) -> NewReno/SACK TCP
  - “satellite” connection” (long RTT, errors) -> optimized TCP (Hybla)
- Advantages due to “splitting” alone
  - “RTT unfairness” problem removed
    - The terrestrial connection is no more penalized by long RTT
- Hybla (i.e. a TCP optimized for satellite) advantages
  - High performance on the satellite channel
    - Fast (re-)opening of cwnd
  - Sender side only modifications

TCP Hybla

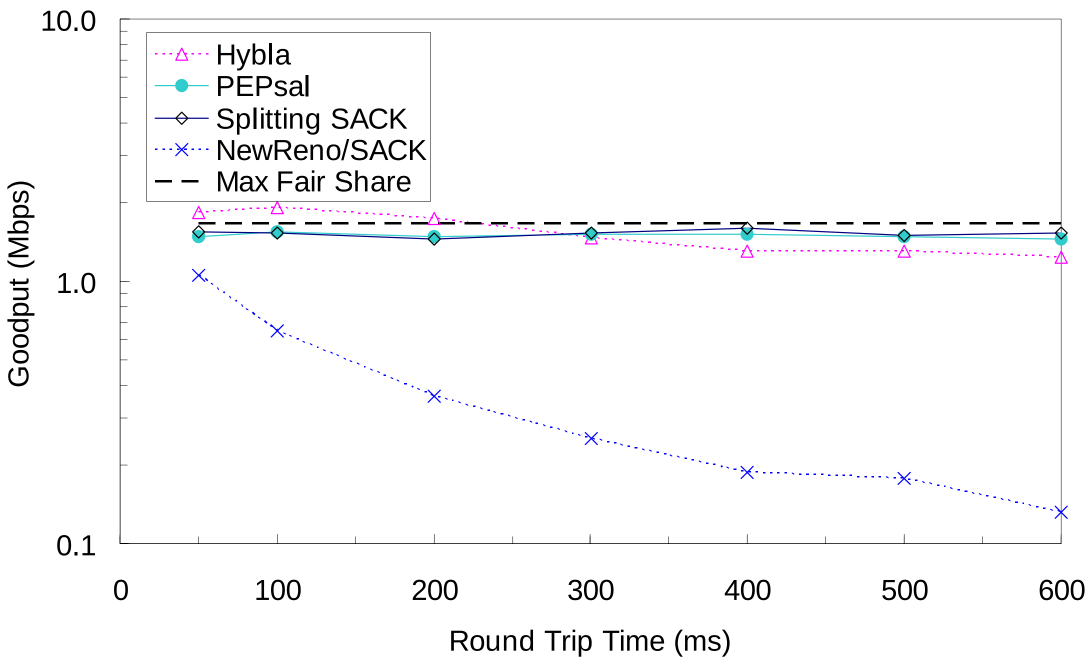
- Design aim
  - To counteract the negative effects of long RTTs
    - RTT unfairness, poor performance at start-up, slow cwnd re-opening after recovery phases
  - Sender side only modifications
- TCP Hybla components
  - Hybla modified cwnd Congestion Control (CC) algorithm
  - SACK
  - Timestamps
  - Pacing
  - Hoe's initial bandwidth estimation
- In Linux official kernel (also available in NS-2)
  - Hybla CC included in Linux standard kernel from 2.6.13
  - Pacing and Hoe's require additional Multi-TCP patch (sf.net/projects/multitcp)

TATPA testbed layout



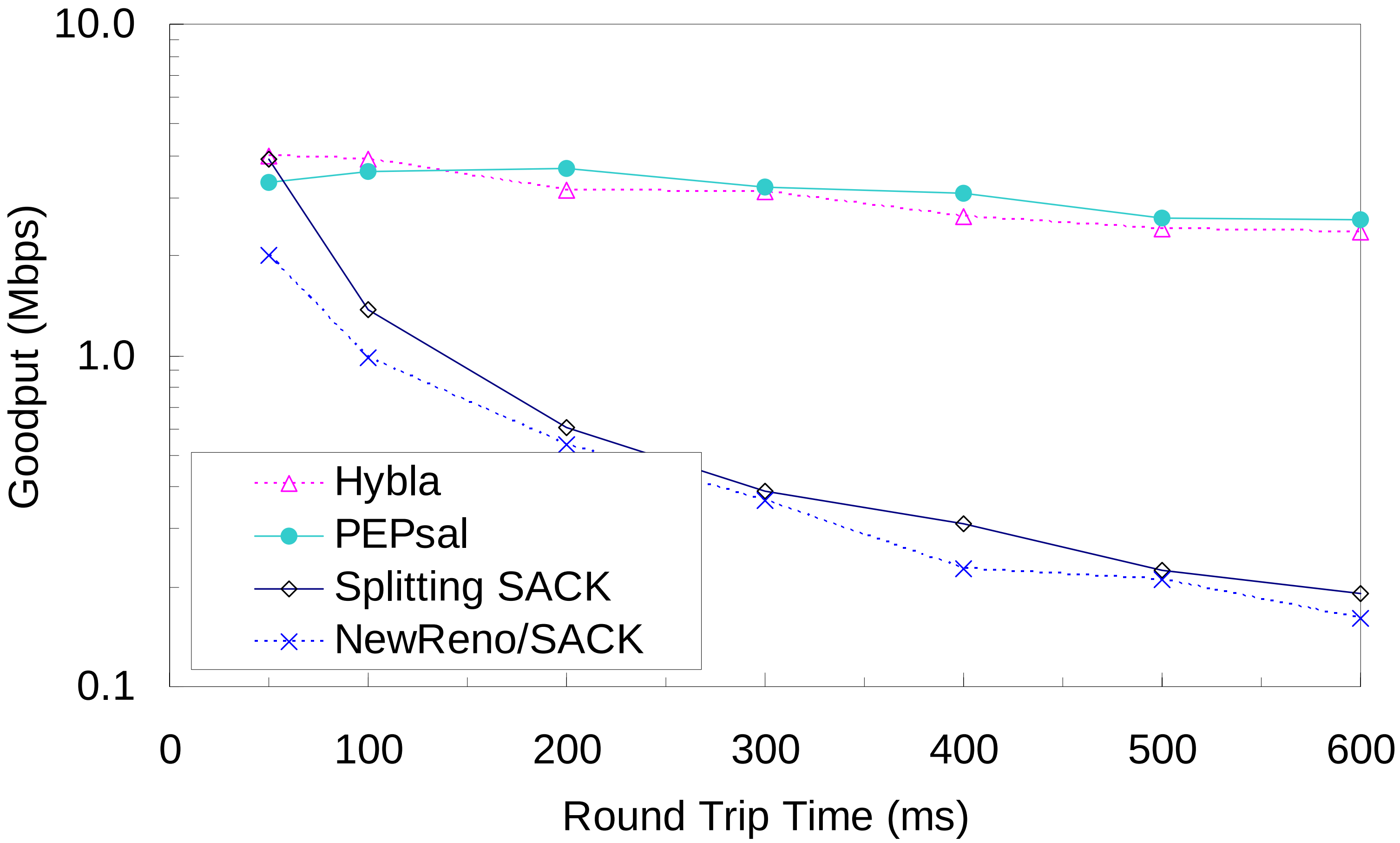
Performance: congestion on the R1-R2 bottleneck

- Goodput of a satellite connection vs. its RTT
  - 5 background wired connection (RTT=25 ms) active; PER=0% on the satellite channel



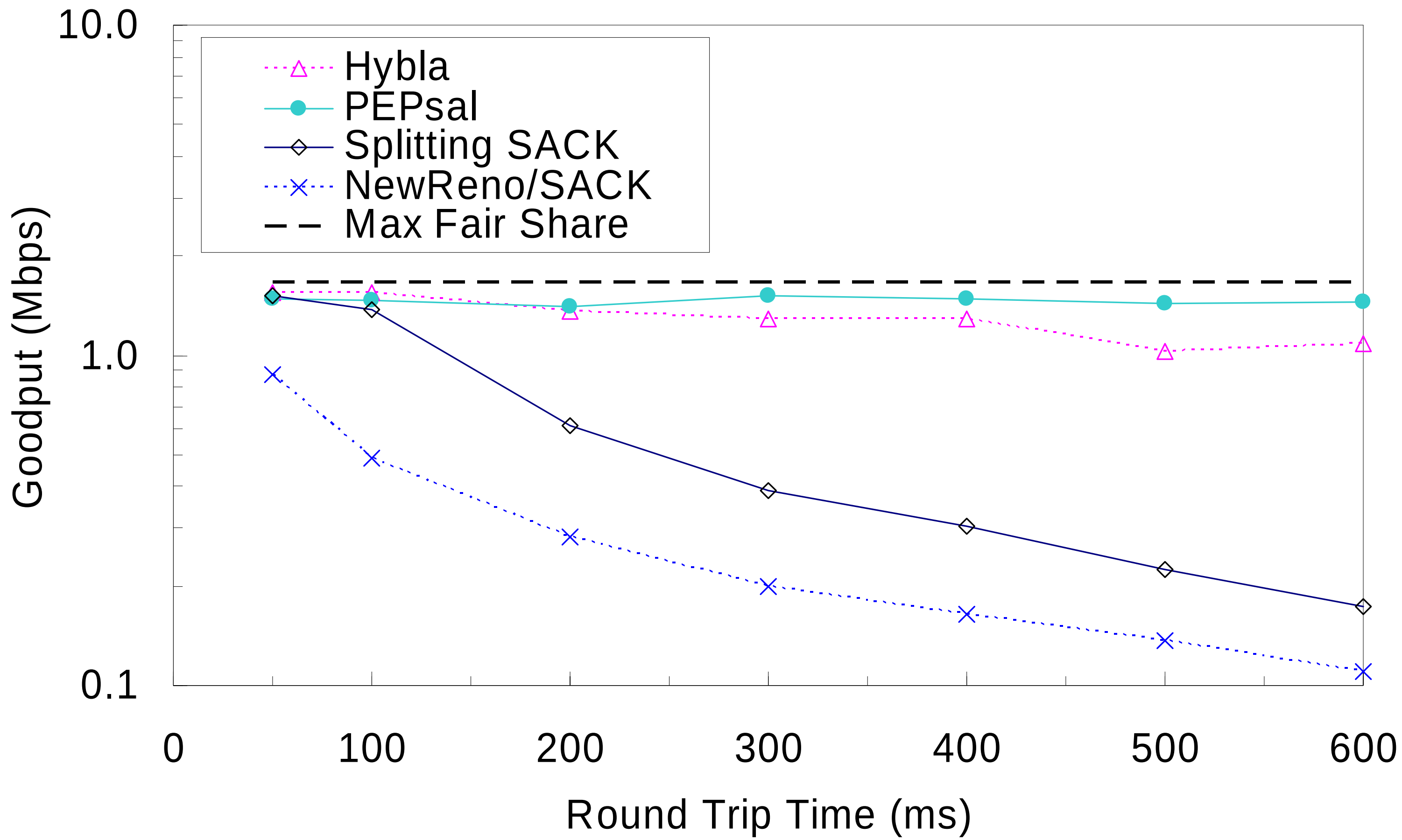
Performance: PER on the R1-R2 bottleneck

- Goodput of a satellite connection vs. its RTT
  - no background wired connection; PER=1% on the satellite channel



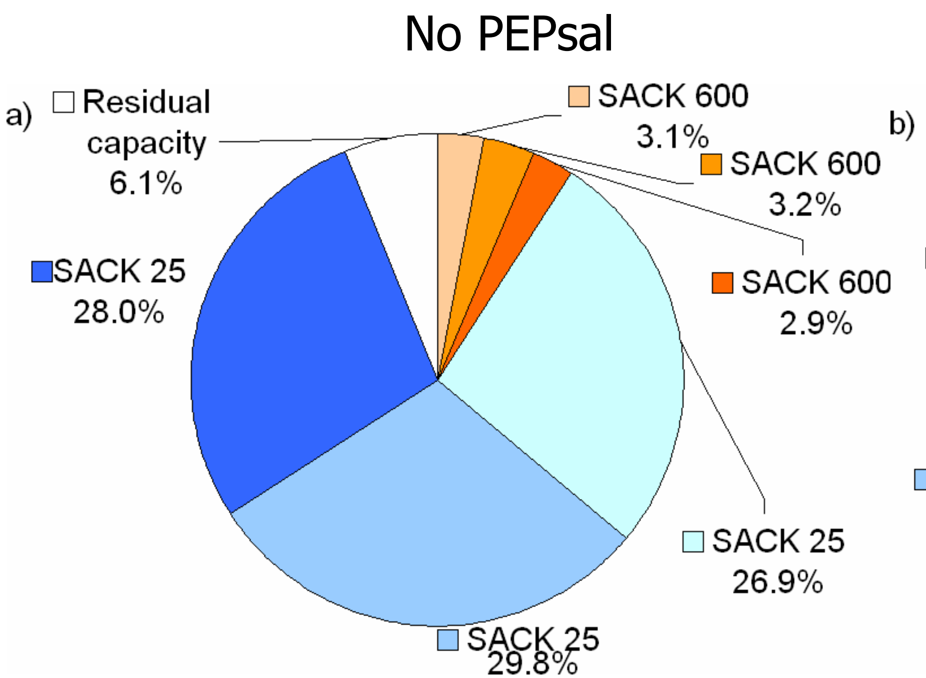
Performance: congestion on the R1-R2 bottleneck & PER on sat

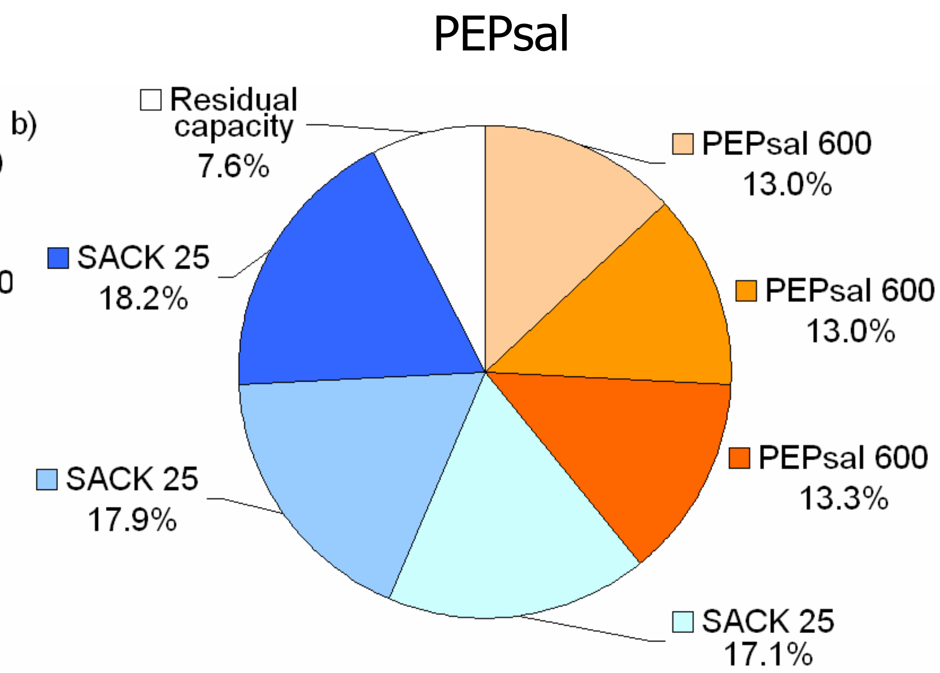
- Goodput of a satellite connection vs. its RTT
  - 5 background wired connection (RTT=25 ms) active; PER=1% on the satellite channel



Performance: friendliness & fairness

- R1-R2 bottleneck share in a heterogeneous environment
  - 3 Terrestrial (in blue)
  - 3 satellite (in orange)





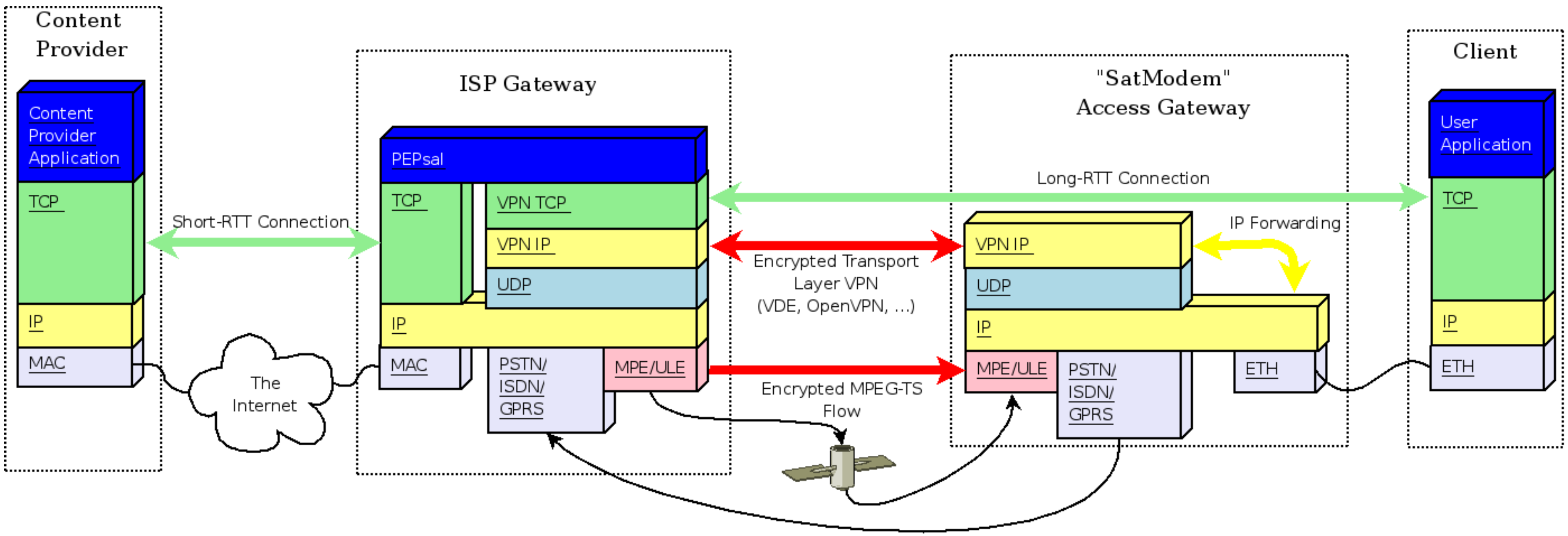
Integrated vs distributed PEPs

- Integrated pros
  - Single-box: neither HW or SW or specific OS required at user side
    - Inexpensive
    - Easy to maintain, to tune, to update (e.g. future enhanced TCP versions)
- Integrated cons
  - Non-standard transport protocol are not allowed on the sat segment (i.e. proprietary solutions forbidden)
  - Mainly asymmetric
    - Symmetric configuration possible but
      - either provide only "splitting "advantages
      - or optimized TCP protocol adopted also on user side

PEPsal "service"

- At first developed for academic purposes, then adopted by a US one-way satellite broadband Internet provider to offer Internet access in US & Latin America
  - Was maintained by Daniele Lacamera
- Two different implementations of the same concept (transparent, integrated, TCP-Splitting, Hybla)
  - The UoB implementation of PEPsal (Latin America service)
    - More suitable for testing (hundreds of concurrent users)
    - Works on all TCP connections
  - Squid (configured as a transparent proxy) + Netfilter + TCP Hybla (US service)
    - More scalable (thousands of concurrent users)
    - Works on specific TCP based services (HTTP, FTP..) only
- Lesson learned from real service:
  - Performance is affected by the whole scenario (DVB encapsulation, flows priority, multicast traffic) which may require customization of the protocols in the stack

Things are never as simple as they appear…(actual on service scheme)



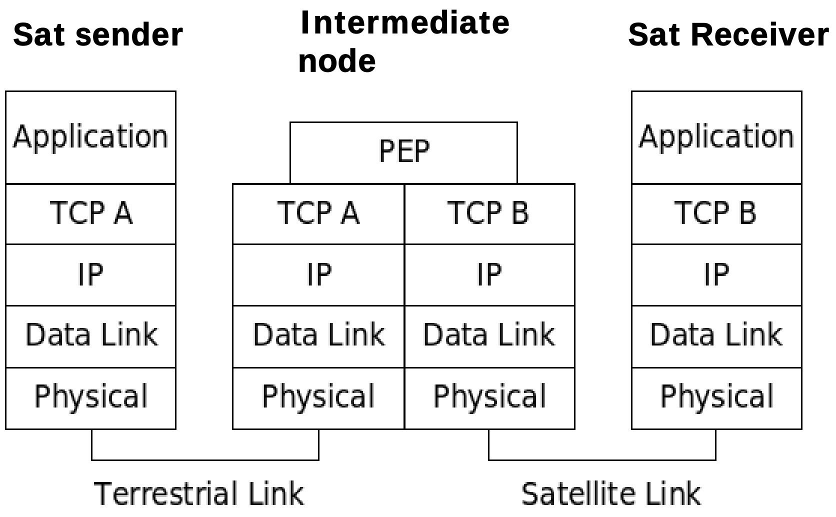
PEPsal conclusions

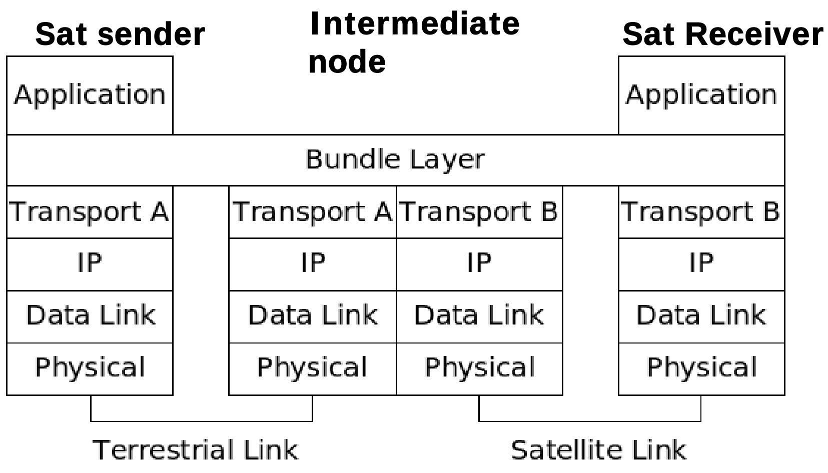
- PEPsal advantages
  - Good performance
  - No license fees (free software)
  - Easy maintenance
  - Extensible
- Integrated solutions should be considered in future PEP standards

Future research directions: DTN on satellites

- DTN aim: to support communication in challenged environment
  - "Bundle layer" added to the stack
  - "Custody transfer" mechanism (optional)
- Reliability
    - Reliable transport layers in charge of reliability only inside homogeneous segments
    - End-to-end reliability in charge of application assisted by bundle layer ACKs
- Advantages
  - Possibility to make use of optimized Transport protocols on different segments
  - Elegant solution (no end-to-end semantic violation)
- Disadvantages
  - Require adoption of bundle protocol at end nodes and at DTN routers
  - Overhead due to the bundle layer (actually, limited as shown later)
- DTN architecture considered in our experiments aims at exploiting
  - Maximum commonality with the PEPsal architecture

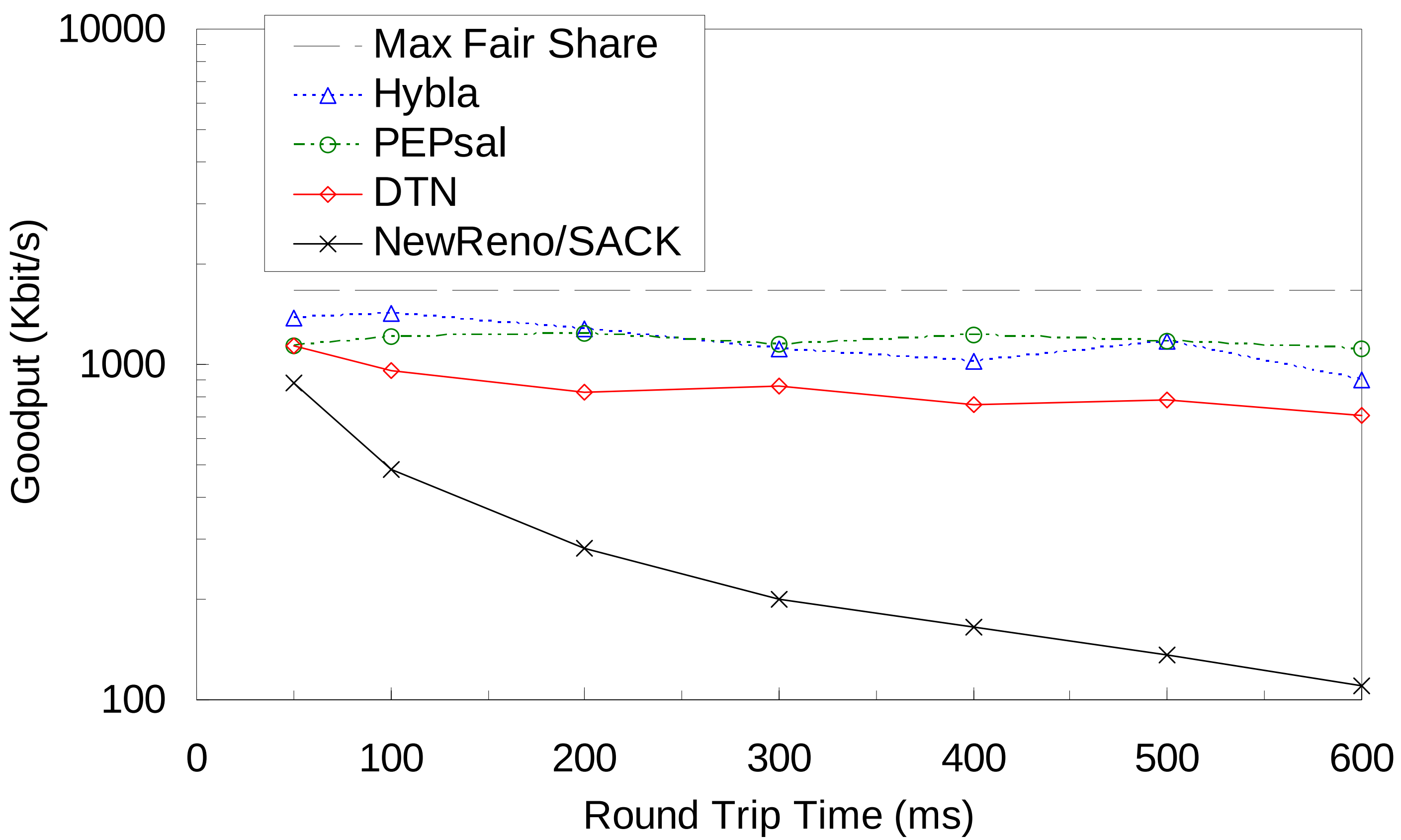
PEP & DTN architectures



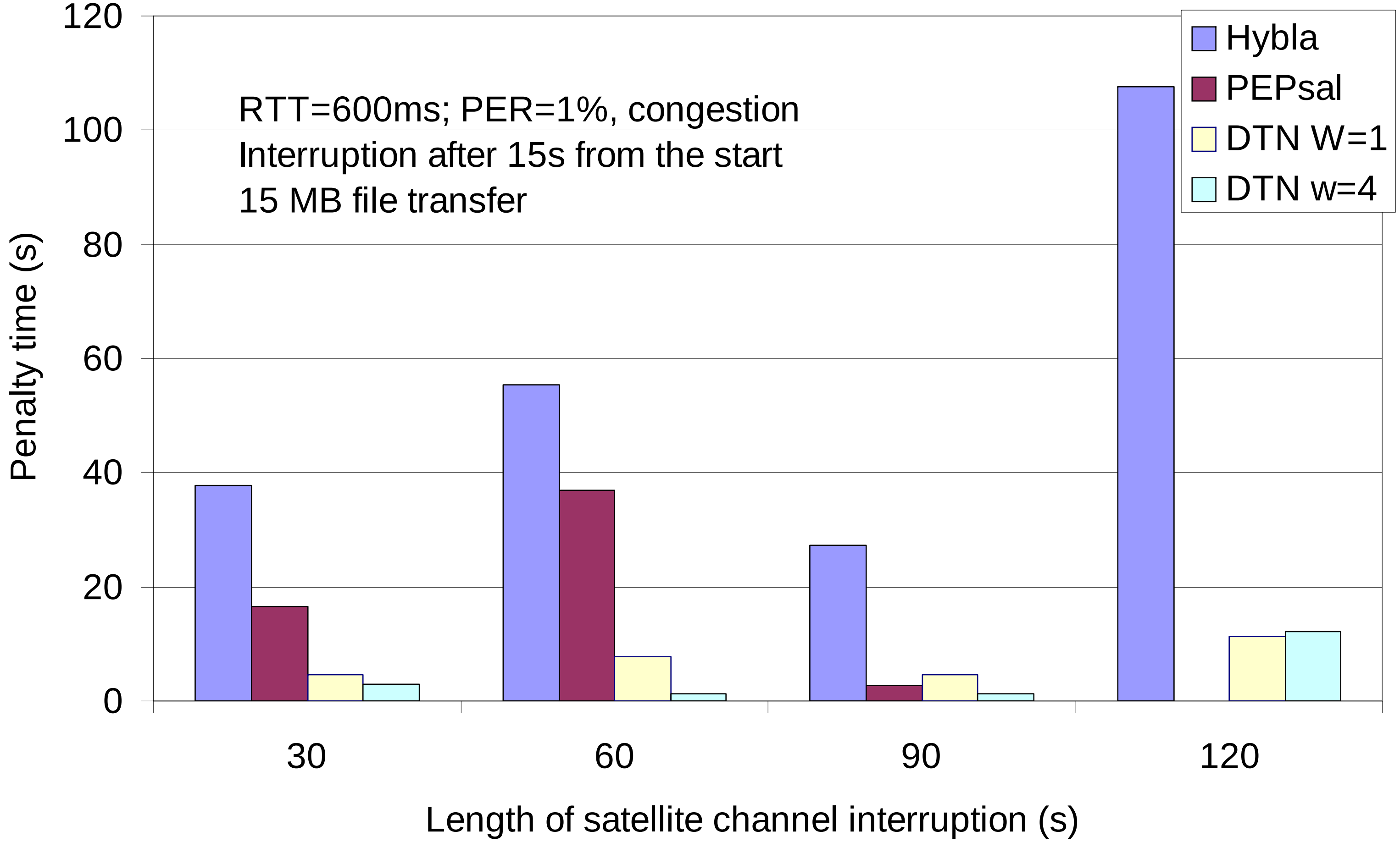


Performance in presence of both congestion and link losses

- Goodput of satellite connection vs.RTT
- PER=1% on the sat. link; 5 background wired connections (RTT=25 ms)



Perfromance on disruptive channels (short interruptions)



Conclusions

- PEPs
  - Effective mature solutions
  - Need standardization
- DTNs
  - Promising medium/long term solution
- UoB open to research collaboration!