spectracom
Synchronizing Critical Operations

PTPv2-based Network Path Load Estimate

Gilles Boime,
Chief Scientist
Email: gilles.boime@spectracom.orolia.com

ITSF, Nov. 2013

Agenda

- Motivation
- Methodology
- Path delay vs network load
- Load estimate
- Use cases: video streaming application
- Conclusion and further study

Work Team

- This work has been done in cooperation team:
  - Pantelis Frangoudis
  - Adlen Ksentini
  - Yassine Hadjadj-Aoul
  - Gilles Boime

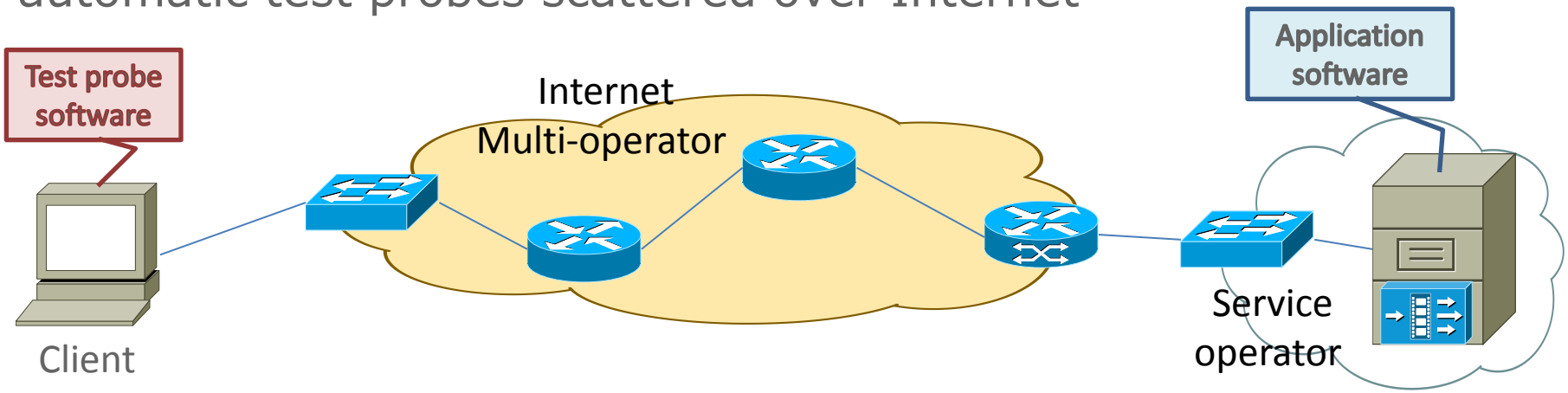
INRIA Rennes-Bretagne Atlantique

Spectracom

- The network Quality of Experience is tested by Telecom service third party qualification expert
- This work is partially funded by French ministry for Industry and Paris - Ile de France Region

Motivation: the foundation

- Service providers monitor the Quality of Experience with automatic test probes scattered over Internet

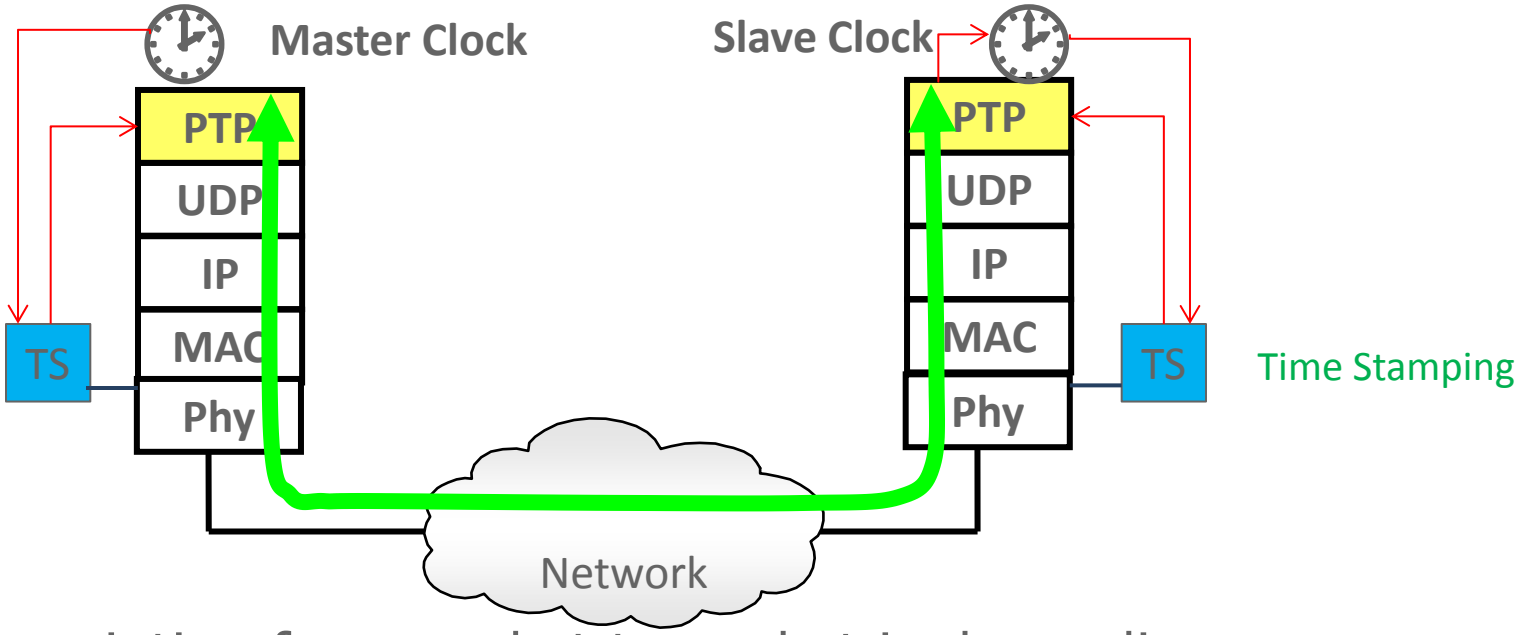


- Need to separate Service operator / application capability from Telecom operator

- Find a way to evaluate at service level the contribution of network path availability to the Quality of Experience provided by the Telecom Operators

What can PTP do?

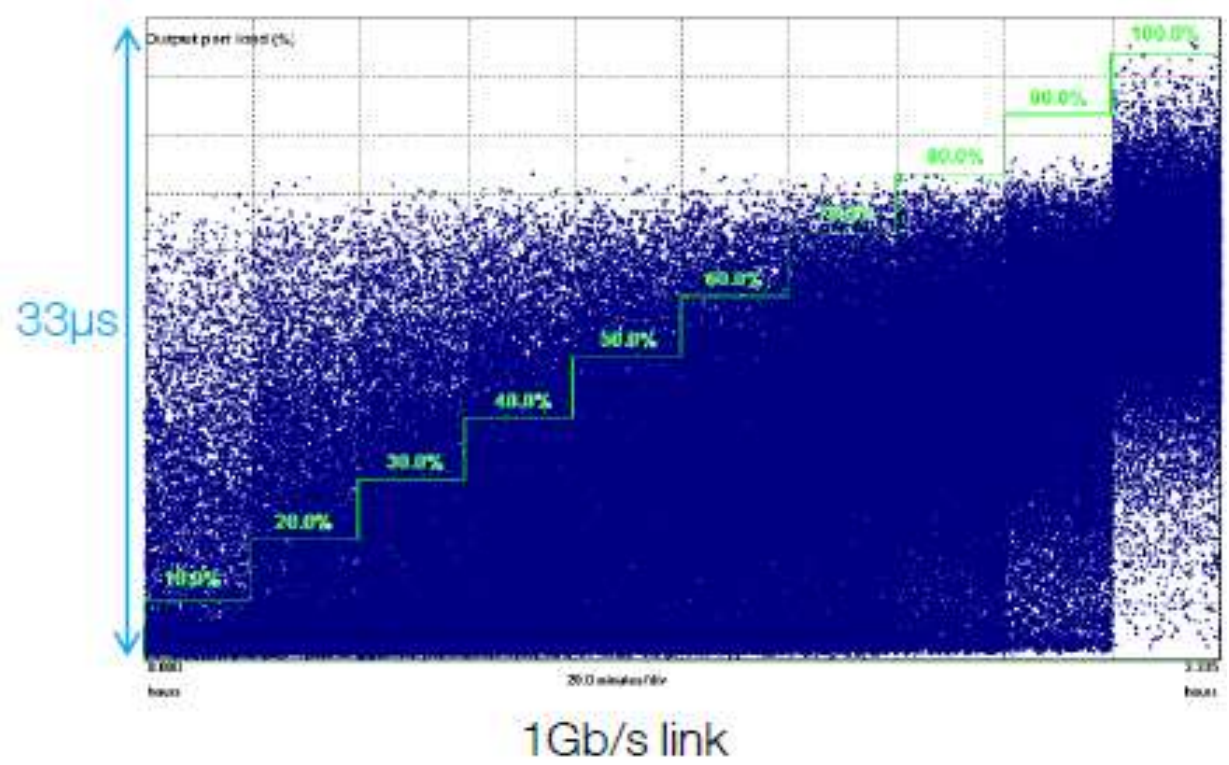
- Timing packet accurate hardware measurement can deliver > ×100 improved measurement vs software applications



- Delay variation from packet to packet is depending on:
  - Route switching
  - Route load, NE queues store and forward

PTP packets delay vs load

- The probability of a packet delay to reach some value is depending from load on the network path



From Orange Presentation at ITSF2011, S. Jobert

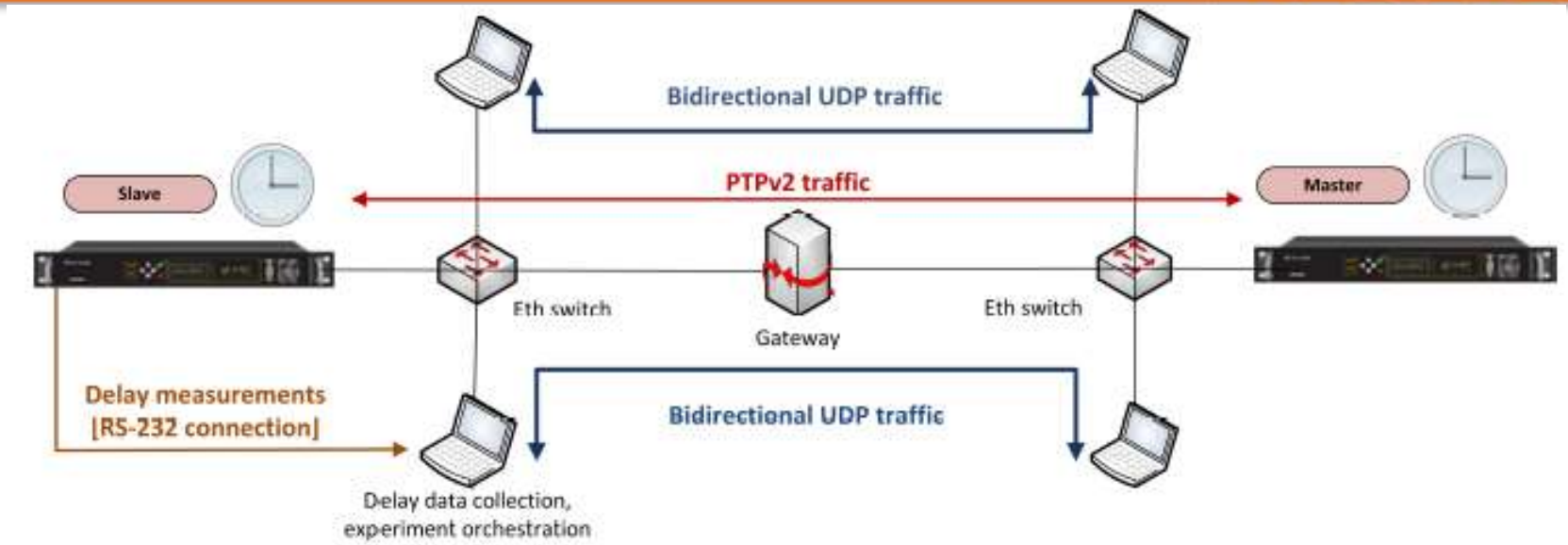
1Gb/s link

- We can be able to inverse function to derive load from path delay

Methodology: Delay-based network load estimate

- Assumptions:
  - Path delay is a function of network load
  - This function depends on the network topology and traffic characteristics
- Principle:
  - Observe: Measure PTP timing packets time stamping behavior (Path Delay)
  - Learn: Model behavior vs. network load conditions
  - Estimate: Apply model to predict network conditions
- Methodology:
  - 1- Monitor traffic and derive traffic model
  - 2- Generate synthetic traffic & measure delay
  - 3- Fit models to delay measurements & derive expression load = F(delay)
  - 4- Implement model into estimation tool

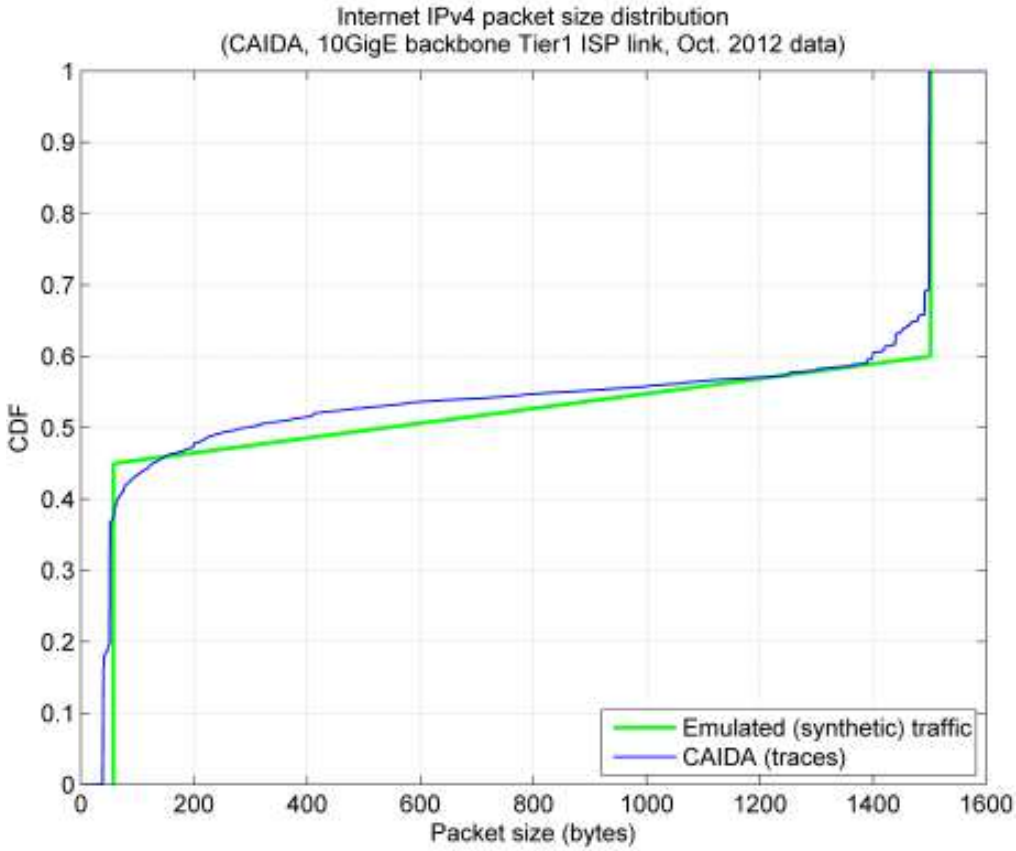
Methodology: Testbed setup



- **PTPv2 implementation:**
  - Spectracom SecureSync PTPv2 devices, Unicast communication, 2-steps mode
  - Protocol implementation at the NIC-level with hardware time stamping ±4ns resolution
- **Testbed capabilities:**
  - 100 Mbps Ethernet interconnection
  - Linux-based gateway separating master-slave LANs
  - 2 TX – RX pairs of hosts to generate workload up to 2× connection capability

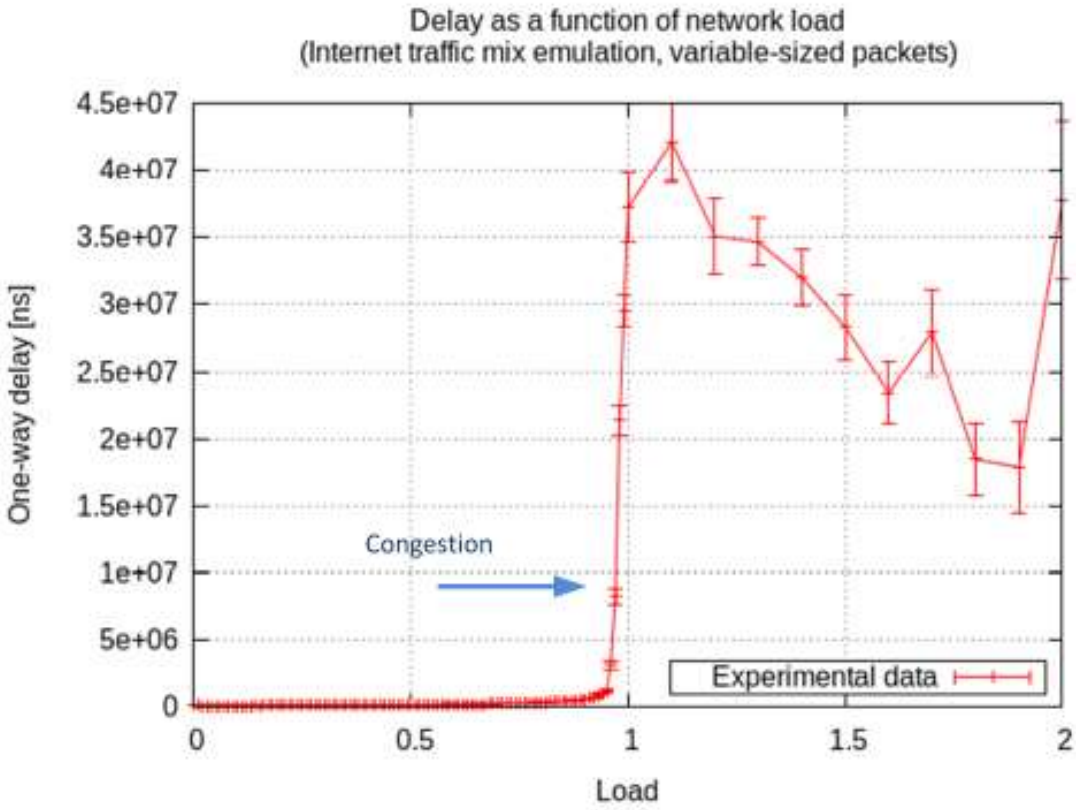
Methodology: Workload generation

- Emulate an Internet traffic mix:
  - Synthetic traffic based on ISP backbone link traces (CAIDA, Oct. 2012)
  - Bimodal packet size distribution (small and large packets prevail)
- Measurements:
  - Bidirectional UDP flows across the gateway
  - Achieving target load with variable-size packets
  - Load value normalized with testbed capacity (e. g., 120 Mbps → 1.2)



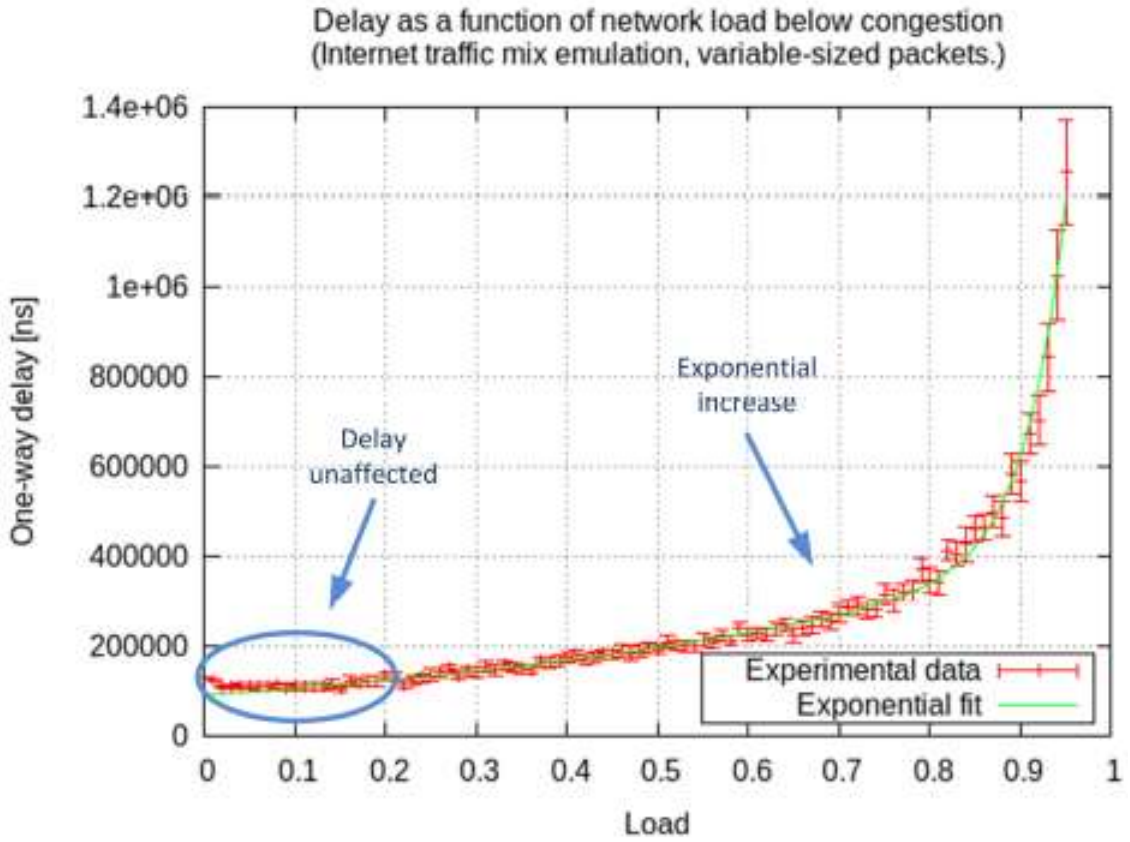
Path delay vs. network load
Path delay vs. load dependency

- Experiments:
  - Load values: 0.0 → 2.0
  - For each load value, measure avg. path delay over ≈ 500
- Low load (< 0.2): Path delay unaffected
  - Identify low-load delay threshold
- Congestion (> 0.95): Increase by order of magn
  - Identify congestion point
- Below congestion: Exponential delay increase
  - Fit an exponential sum function to data



Path delay vs. network load
Path delay before congestion

- Packet distribution emulated:
  - 45% of 58 byte
  - 40% of 1500 byte
  - 15% uniformly distributed between 58 – 1500 byte
- Determination coef $R^2 = 0.998$

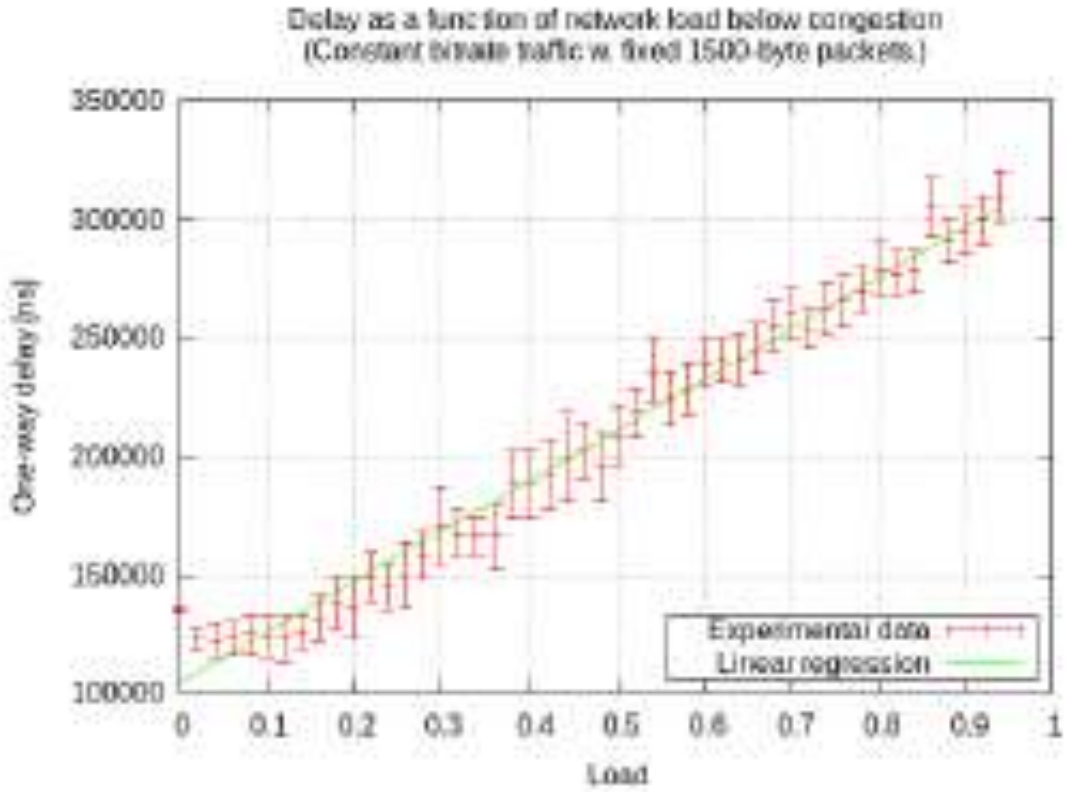


- **Behavior depends on network topology and traffic pattern**

Path delay vs. network load
Path delay vs. load dependency

- With constant bit rate traffic of fixed 1500 bytes packets linear behavior between low-load threshold and congestion point

- Determination coef $R^2 = 0.9734$

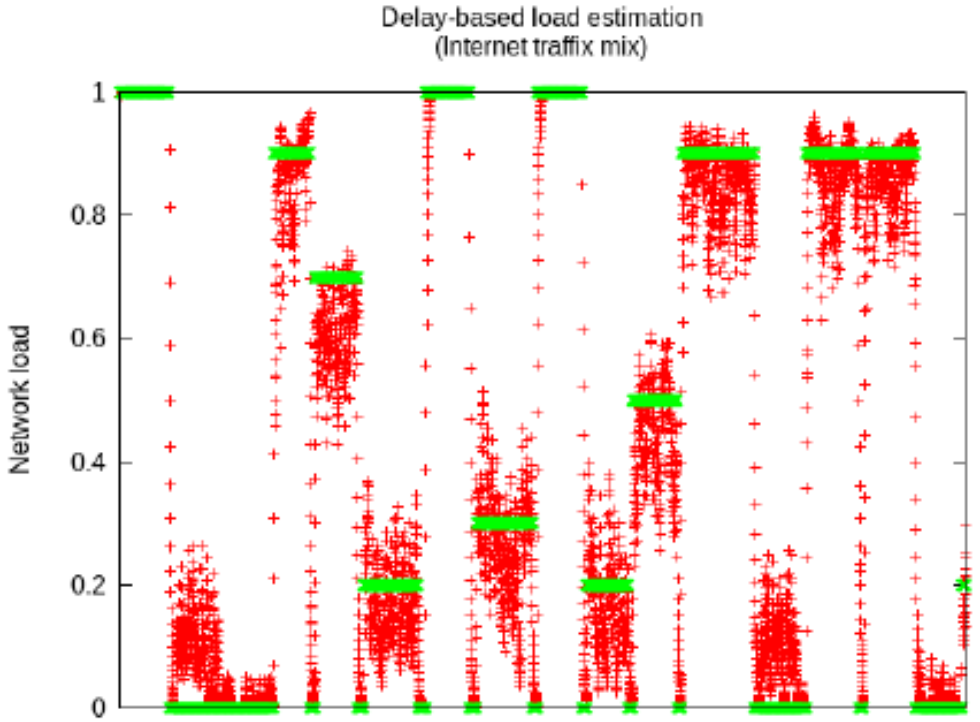


Load Estimate: Algorithm

- Phase 1: Parameter derivation
  - D_{low}: low-load threshold
  - D_{high}: Congestion threshold
  - Select function of delay vs. load
- Phase 2: Real-time operation
  - Maintain window of k last delay samples
  - Maintain weighted moving avg. of estimated load
  - Fore each delay sample:
    - Slide window, get window average ($\bar{d}$)
    - If $\bar{d} < D_{low}$ then $L_{sample} := 0.0$
    - Else If $\bar{d} \geq D_{high}$ then $L_{sample} := 1.0$
    - Else $L_{sample} :=$ min(mapDelayToLoad($\bar{d}$), 1.0)
    - Update load estimate: $L := wL + (1-w)L_{sample}$
- Optimal values for k and w
  - Generate random load patterns
  - Measure estimation accuracy for parameter combination
  - Select values which minimize estimation RMSE

Load Estimate: Evaluation

- Random load patterns test
  - Load values: 0.0 → 2.0
  - k = 1, w = 0.87
  - Measured RMSE = 10.8%



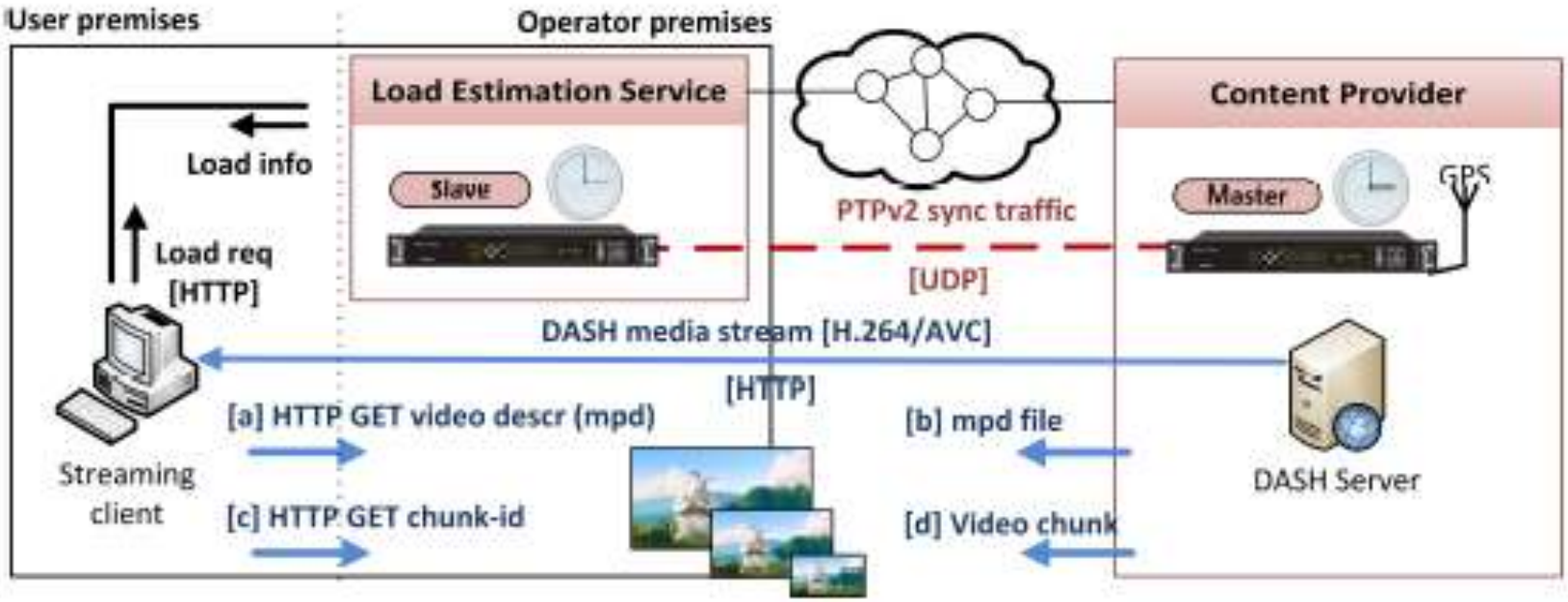
Use Case:
Load-aware adaptive video streaming

- Network environment architecture
  - Operator / content provider installs PTP equipment close to user premises
  - Master clock collocated with media origin server
  - Load estimation service accessible to client application
  - Purpose: Monitor conditions on the media path
- Application
  - Adaptive video streaming over HTTP (DASH)
  - Content available at different qualities / bit rates
  - Receiver-driven approach
  - Client query Load Estimation Service (LES) for real-time load information

Use Case:
Load-aware adaptive video streaming

- Adaptation Scheme
  - Satisfy QoE and network capacity constraints
  - Operator-defined max load threshold (e.g. 0.5)
  - Increase bit rate when load is above threshold
  - Decrease bit rate otherwise
  - Limit the switching of bit rate to avoid rockiness of control
- Alternate capabilities
  - Other adaptation scheme are possible
  - Straightforward to extend to SVC support (Scalable Video Coding for H.264 /MPEG-4 AVC)

Use case: Implementation



- Load Estimation Service
  - Accessible using HTTP API
  - Collocated with slave clock
- Video streaming
  - H.264 / AVC
  - LES queries and quality adaptation built into VLC DASH plugin

Conclusion & Further studies

- Conclusion
  - Proposed delay measurement-based to evaluate load on the network path.
  - Use PTP timing packet accurate measurement and protocol association.
  - Under specific learning step, design, implement and evaluate a LES
  - Integrates a LES in an adaptive video streaming architecture for monitoring streaming content.
- Further study
  - Develop other features as load indicator protection (delay variation, packet loss) against route switching or load structure change.
  - Validate the relevance of the estimate regarding other QoE metrics about end to end service.

Conclusion

- **Accurate time is an enabler of new disruptive services on the Net**

- *Open your mind, the horizon is not a frontier*