An Experimental Investigation of the Congestion Control Used by Skype VoIP

L. De Cicco, S. Mascolo, V. Palmisano

Politecnico di Bari, Dipartimento di Elettrotecnica ed Elettronica

5th International Conference on Wired/Wireless Internet Communications

Coimbra, Portugal, 23-25 May 2007

WWIC2007

Outline

1. Motivations
2. Experimental testbed
   - The testbed
3. Experimental results
   - One Skype flow
   - One Skype flow with concurrent TCP connections
   - Two Skype flows sharing the bottleneck
4. Conclusions

Transport of Multimedia flows

- The convergence of multimedia services (VoIP, video on demand, video conference) has opened the door to new challenges
- The efficient transport of multimedia flows is still an open issue
- It is not yet clear what will be the impact of VoIP traffic on the stability of the Internet (*"congestion collapse*?")
- Some protocols designed for the transport of multimedia flows are:
    - TCP Friendly Rate Control (**TFRC**): it is currently discussed within the IETF
    - RAP, TEAR

Goals of the work

Does Skype harm network stability?

- Skype is by far the most used VoIP application generating a large amount of traffic
- Skype uses UDP flows for VoIP transport: investigate their characteristics (Skype is a closed source application)
- There is no evidence that Skype flows would impact the stability of the best-effort Internet

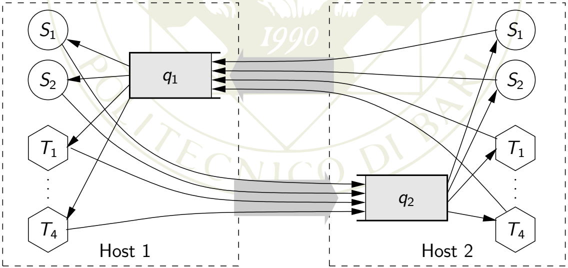
The goals of the work

1. Are Skype flows inelastic?
2. How does Skype react to network congestion?
3. How does Skype adapt the sending rate to match the available network bandwidth?
4. Is Skype fair with other concurrent Skype and TCP flows?

Outline

Setup

- A local testbed has been set up using a measurement tool we have developed (`ipq-shaper`)
- All packets generated from Skype application have been routed to the ingress queues $q_1$and q_2
- Delays, available bandwidth and buffer size of each queue can be set by the user



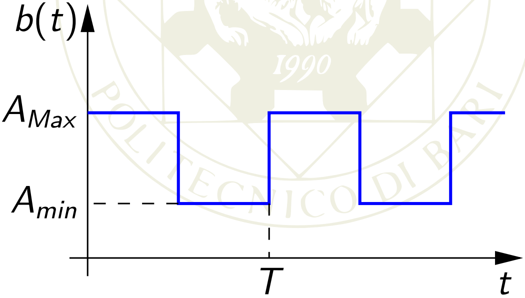
Traffic generation and measurement

- We have installed on each host Skype (S_1 and S_2) and `iperf` (T_1,..., T_4) in order to generate TCP flows
- We collected logfiles measuring goodput, throughput and loss rate, by tracing the per-flow arriving and departing traffic from each queue
- The RTT of the connection is set to 100 ms and the queue size is set equal to the bandwidth delay product.
- Skype flows are generated using always the same audio sequence by hijacking audio I/O in order to perform reproducible experiments

Investigating the Skype congestion control

Methodology

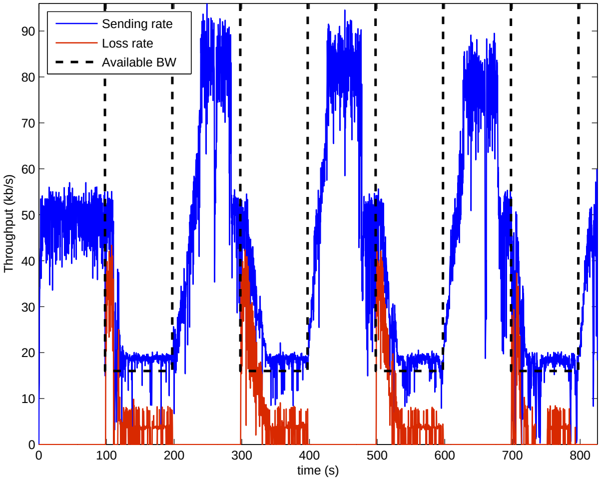
- We have considered step-like time-varying available bandwidths
- Using square-wave available bandwidths (duty cycle is 50%) characterized by different periods we tested:
  - the Skype capability to match the available bandwidth
  - the transient time required for the matching (responsiveness)



Outline

Square wave available bandwidth (period 200 s)

How Skype sending rate reacts to changes in the available bandwidth

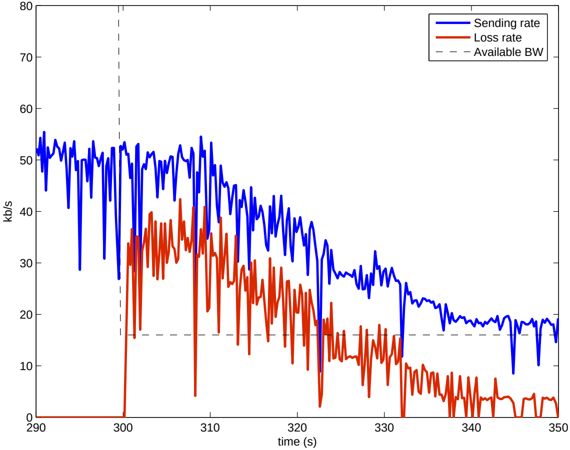


Set-up: square-wave available bandwidth (A_{Max}=160 kb/s, A_{min}=16 kb/s, $T = 200\,s$)

- The loss rate decreases to $<$ 10kb/s after a transient
- The Skype flow is elastic and it is able to match the available bandwidth

Square wave available bandwidth (period 200 s)

Zoom around the bandwidth drop at $t = 300s$



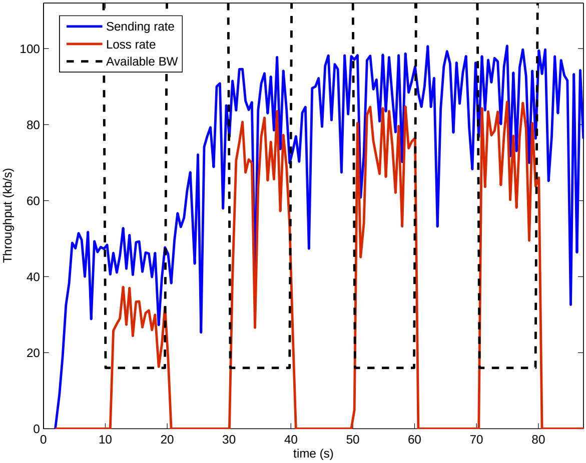
- Skype adapts its input rate to match the available bandwidth in around 40s.
- The input rate throttling seems to be triggered by the increased drop rate
- The algorithm responsiveness is not satisfactory

Question

How does the protocol behave when bandwidth changes happen more rapidly?

Square wave available bandwidth (period 20 s)

How does Skype sending rate react to sudden changes of available bandwidth?

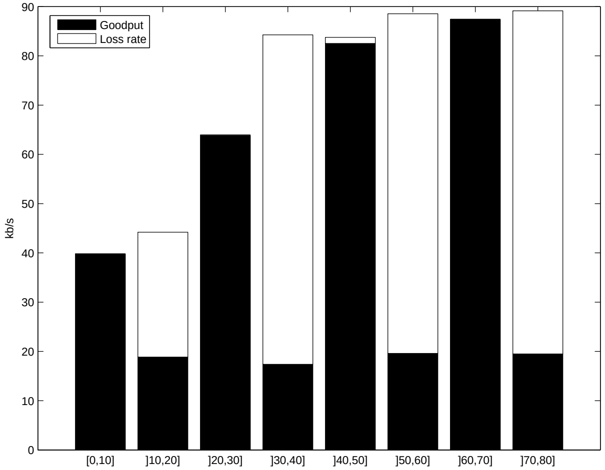


Set-up: A_{Max}=160 kb/s, A_{min}=16 kb/s, $T = 20\,s$

- The input rate remains quite unchanged
- Skype is not able to follow the sudden bandwidth reductions, provoking consistent losses.
- Skype flows are not able to avoid congestion in such scenario because of the long transient times

Square wave available bandwidth (period 20 s)

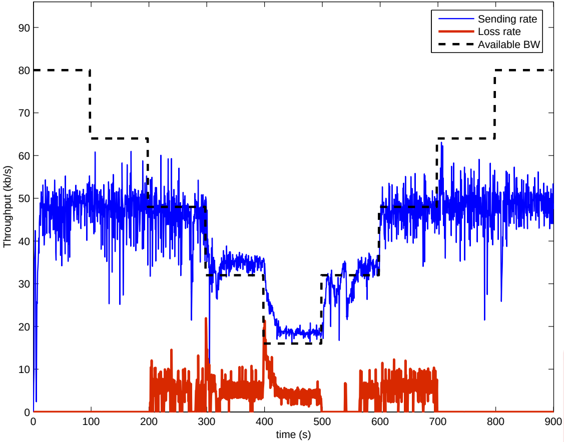
Goodput and loss rate during time intervals at constant available bandwidth



In the time intervals characterized by low available bandwidth Skype flow suffers an high packet loss rate which may not guarantee perceived quality.

Skype flow with variable bandwidth

How does Skype sending rate react to small step-like increases/decreases of available bandwidth?



Set-up: 16 kb/s bandwidth drops/increases every 100 s. $b(t) \in [16, 80]\ kb/s$ (min/max measured rates).

- What is the granularity of the input rate produced by Skype flows?
- Is it able to adapt to small step-like variations?

The input rate is able to contain packet losses, however it does not follow the available bandwidth (in the interval $[700, 900]$)

Outline

Skype and TCP flows sharing the bottleneck

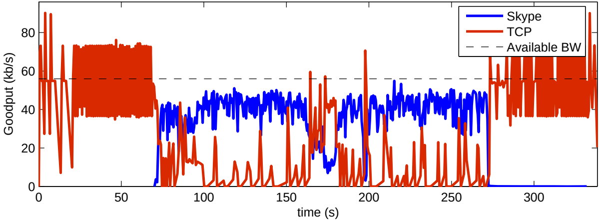
TCP Congestion Control

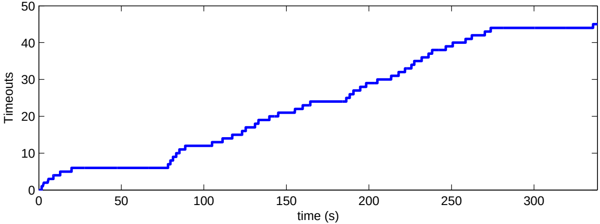
- TCP is still the most used transport protocol and it is the main driver of the stability of the network
- It is a loss based congestion control algorithm
- Congestion is detected when three duplicate ACK are received

Skype congestion control

- The response to congestion is slow (˜40 seconds to adapt to available bandwidth)
- **How do the two congestion control algorithms interact when both Skype and TCP flows are accessing the bottleneck?**

One concurrent TCP flow (constant bandwidth)

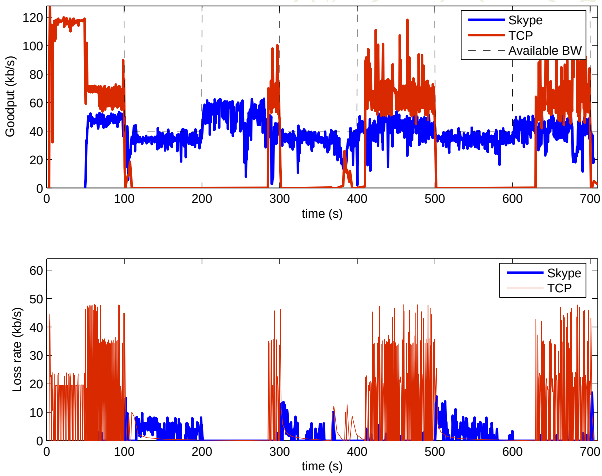




Set-up: Constant link capacity of 56 kb/s. Skype call starts 70 s later than the TCP flow.

- When the Skype flow enters the link it causes a very large number of timeouts in the TCP flow.
- Goodput of TCP flow is near to zero when Skype flow is on.
- Skype does not share the available bandwidth fairly

One concurrent TCP flow (square wave bandwidth)

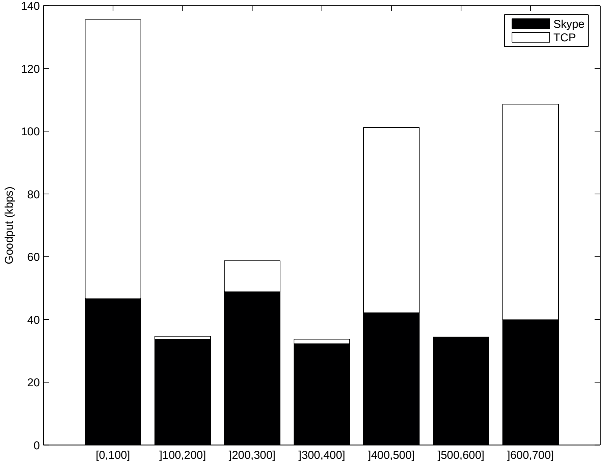


Set-up: A_{Max}=160 kb/s, A_{min}=50 kb/s, $T = 200\,s$.

- **How do the two protocols interact when increases and decreases take place?**
- When the available bandwidth is low the TCP connection doesn't get any share.
- TCP suffers of an high number of timeouts

One concurrent TCP flow (square wave bandwidth)

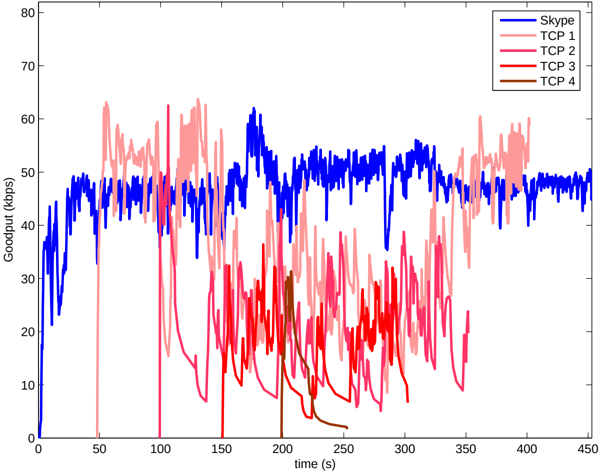
Goodput of Skype and TCP flows in time intervals where the available bandwidth is kept constant



Observation

When the available bandwidth is low, the TCP flow is not able to get any share, and its goodput is close to zero.

Four concurrent TCP flows

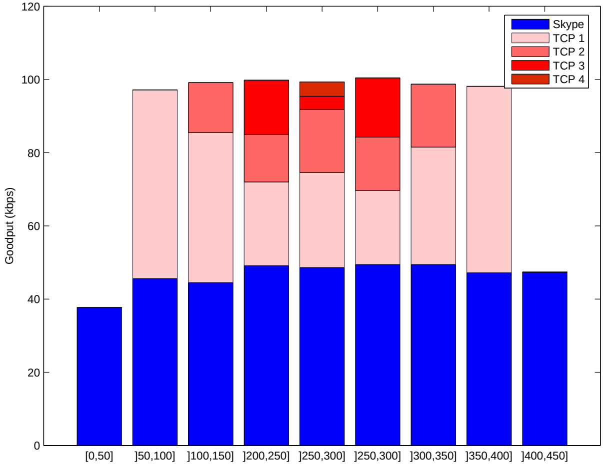


Set-up: $b(t) = 120\ kb/s$
In the first half it is started one TCP connection each 50s, in the second half it is turned off one TCP connection each 50s.

- Skype doesn't adapt its sending rate when a new TCP flow joins the bottleneck
- TCP flows adapt their rate in order to avoid congestion on the link

Four concurrent TCP flows

Goodput of the flows during each time interval



- Skype's goodput is kept unchanged during all the time, while TCP flows share the left available bandwidth.
- **Skype is not responsive when TCP flows join the bottleneck**

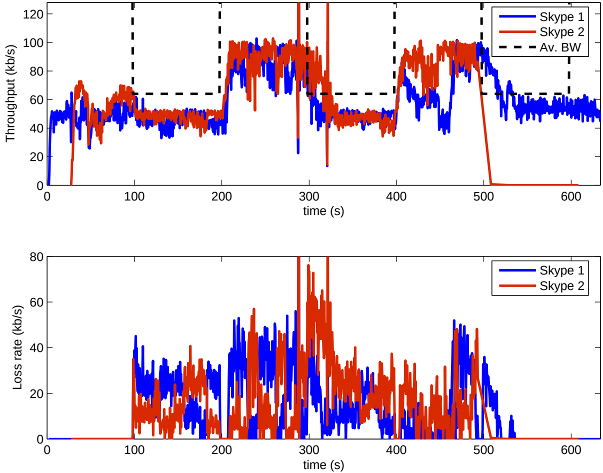
Outline

Two Skype flows sharing the bottleneck

We have seen that Skype congestion control is not TCP friendly, so to conclude the investigation we question about:

- How do Skype flows interact between each other?
- Are they able to avoid congestion on the bottleneck?

Two Skype flows (square wave available bandwidth)



Set-up: Square-wave available bandwidth (A_{Max}=144 kb/s, A_{min}=64 kb/s, $T = 200\,s$). The second Skype calls is placed after 25 s.

- The two flows behave at the manner
- They are not able to avoid congestion provoking consistent losses (up to 80 kb/s)
- Other test have shown that Skype is neither fair

Conclusions

Upside...

Skype implements some sort of congestion control algorithm

...Downside

- The reaction speed of this algorithm revealed to be very slow
- Skype has shown two remarkable drawbacks:
  1. Large packet drop rates during the transients following a bandwidth reduction
  2. Unresponsive behaviour when coexisting with responsive flows such as TCP
- When more Skype calls are established on the same link, they are not able to adapt their sending rate to match correctly the available bandwidth (risk of network congestion collapse)

Further work agenda

- Investigate Skype's behaviour over lossy links?
- Experiment with large number of Skype flows sharing a bottleneck (testbed challenges)
- Skype congestion control algorithm identification using control theory tools (nonlinear switched system)
- Investigate Skype Video congestion control (testbed challenges, how to hijack video?)

Questions?

de' remi facemmo ali
1990
POLITECNICO DI BARI

Any questions?