Safe increase of the TCP's Initial Window Using Initial Spreading

draft-irtf-iccrg-sallantin-initial-spreading-00.txt

R. Sallantin[1] C. Baudoin[2] E. Chaput[1] F. Arnal[2]
E. Dubois[3] A-L. Beylot[1]

[1]IRIT Université de Toulouse [2]Thales Alenia Space

[3]Centre National d'Études Spatiales

February 28, 2014

Outline

Context

- Good performance of last TCP algorithms for long-lived connections

Unfortunately

- Poor efficiency of regular TCP mechanisms for short-lived connections
- Problem even bigger for satcoms because of the long RTTs

Context

- Good performance of last TCP algorithms for long-lived connections

Unfortunately

- Poor efficiency of regular TCP mechanisms for short-lived connections
- Problem even bigger for satcoms because of the long RTTs

90% of web requests are shorter than 10 segments

High level contribution

Initial Spreading concept:

Spread a large amount of data accross the first RTT

Speed the transmission of the first segments

AND

Minimize the impact on the bottleneck link.

Goal: reduce the average latency

High level contribution

Initial Spreading concept:

Spread a large amount of data accross the first RTT

Speed the transmission of the first segments

AND

Minimize the impact on the bottleneck link.

Goal: reduce the average latency

Take the best of 2 TCP mechanisms:

- Increase in the TCP's Initial Window
- TCP Pacing

Increase in the IW

To satisfy 90 % of web requests in 1 RTT

RFC 6928 recommends to set the IW up to 10 segments.

J. Chu,N.Dukkipati,Y.Cheng,M.Mathis, Increasing TCP's Initial Window RFC 6928

Increase in the IW

To satisfy 90 % of web requests in 1 RTT

RFC 6928 recommends to set the IW up to 10 segments.

J. Chu,N.Dukkipati,Y.Cheng,M.Mathis, Increasing TCP's Initial Window RFC 6928

In uncongested network: The fastest solution

Increase in the IW

To satisfy 90 % of web requests in 1 RTT

RFC 6928 recommends to set the IW up to 10 segments.

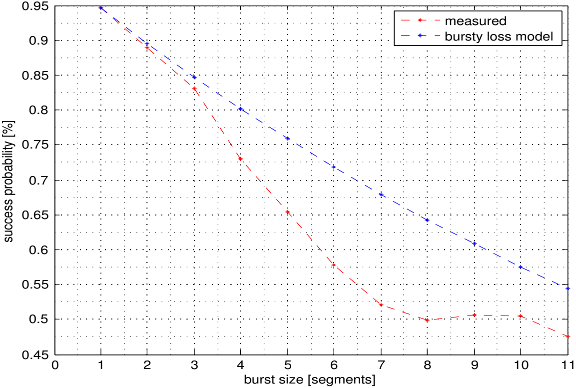
J. Chu,N.Dukkipati,Y.Cheng,M.Mathis, Increasing TCP's Initial Window RFC 6928

In uncongested network: The fastest solution

In congested network: What is the real impact of this initial burst?

Bursts consequences

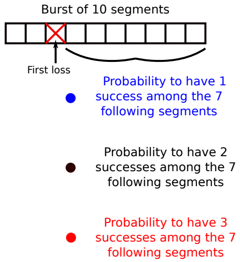
1. Decrease of the success probability
2. Decrease of the efficiency of regular TCP's loss recovery mechanisms
3. Reduce the benefits of a large IW

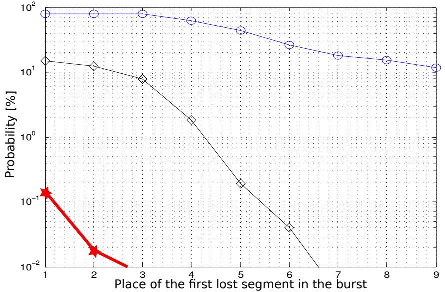


Burst consequences according to the model and real experimentations

Bursts consequences

1. Decrease of the success probability
2. Decrease of the efficiency of regular TCP's loss recovery mechanisms
3. Reduce the benefits of a large IW



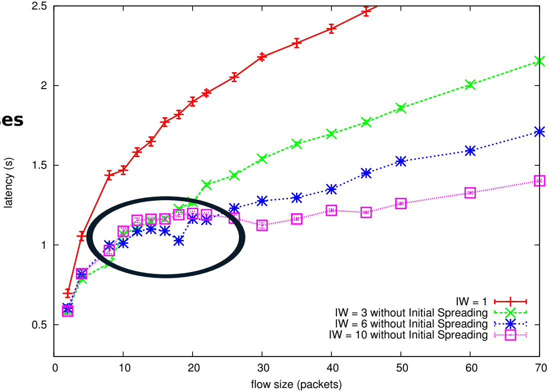


Probability for 1, 2 or 3 segments of the burst to be correctly transmitted when one of the previous segment of the burst has been lost

Bursts consequences

1. Decrease of the success probability
2. Decrease of the efficiency of regular TCP's loss recovery mechanisms
3. Reduce the benefits of a large IW

- **Bursts correlate the losses**
- **Bursts reduce the benefits of a large IW**



Different IWs

Pacing

Pacing aims to prevent the generation of bursts

Concept:

- Spread window transmission over the RTT

Pacing

Pacing aims to prevent the generation of bursts

Concept:

- Spread window transmission over the RTT

Consequences:

- Increases the bit rate by reducing the isolated congestion

  BUT

- Delay the losses, and then, delays the congestion until a potential network collapse

Pacing

Pacing aims to prevent the generation of bursts

Concept:

- Spread window transmission over the RTT

Conclusion:

TCP efficiency needs the loss detection.

=>Pacing downgrades the average TCP performance.

Source: A. Aggarwal, S. Savage, and T. Anderson, Understanding the performance of TCP Pacing, INFOCOM 2000

What we propose

Initial Spreading:

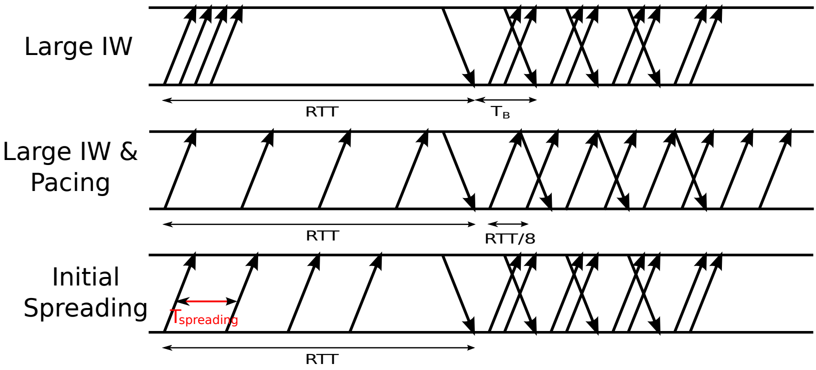
- Spread the IW across the first RTT

  Two possibilities:

  - Variable Spreading: $T_{spreading} = \frac{RTT}{IW}$
  - Bounded Spreading: $T_{spreading} <= T_{max}$

- Let the TCP algorithm continue conventionally after

3 mechanisms



Time diagram for a transmission of 12 segments

Expected behavior

For short-lived connections:

Send a large IW without being affected by bursts

- losses are **independent** in the first RTT
- loss probability is lower
- increase the probability of using recovery mechanisms

Reduce the average latency

Expected behavior

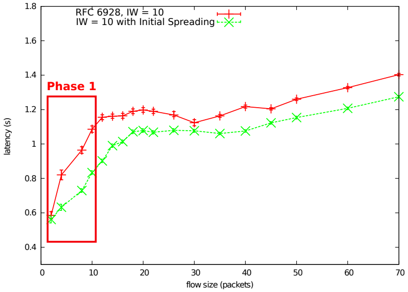
For long-lived connections:

Prevents network overload and synchronization

- As soon as the second RTT, bursts appear
- Losses can continue to indicate the congestion

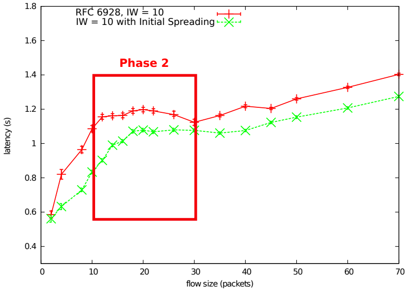
Initial Spreading behavior

1. Phase 1:
   IS reduces the burst impact
2. Phase 2:
   Segments sent (in mini burst of 2) in the 2nd RTT may trigger fast retransmit and recovery
3. Phase 3:
   Congestion avoidance manages the bit rate



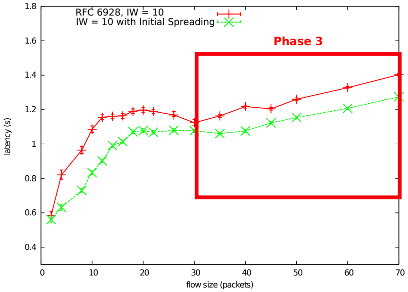
Initial Spreading behavior

1. Phase 1:
   IS reduces the burst impact
2. Phase 2:
   Segments sent (in mini burst of 2) in the 2nd RTT may trigger fast retransmit and recovery
3. Phase 3:
   Congestion avoidance manages the bit rate



Initial Spreading behavior

1. Phase 1:
   IS reduces the burst impact
2. Phase 2:
   Segments sent (in mini burst of 2) in the 2nd RTT may trigger fast retransmit and recovery
3. Phase 3:
   Congestion avoidance manages the bit rate



Considerations

To be efficient, Initial Spreading should take the best of several constraints:

- $T_{spreading}$ MUST be large enough for the losses to be un-correlated
- $T_{spreading}$ SHOULD be the shortest possible
  to not add an un-necessary delay (notably in uncongested network)
- Implementation MUST be light and respects Kernel constraints

Assumption on the losses correlation

Assumption on the losses correlation:

- The minimal spreading depends on the bottleneck throughput
  - Segments spread with $T_{spreading} < \frac{BottleneckThroughput}{MTU}$ will face the same bottleneck buffer state.

Simulations and Experimentation confirm our hypothesis

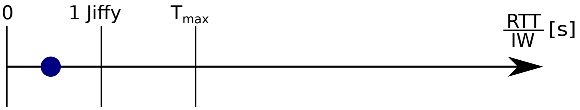
Variable or Bounded Spreading?

- Variable Spreading is related to the RTT measurement
  => add some incertainty
- A Bounded Spreading insures a good losses independence for the IW segments
- A Bounded Spreading eases the implementation

We recommend the use of a Bounded Spreading

Proposal

Our proposal:

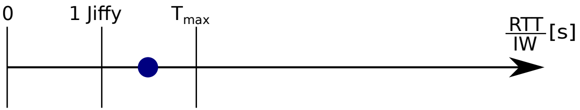


Case $\frac{RTT}{IW} < 1$ Jiffy

Initial Spreading is not used

Proposal

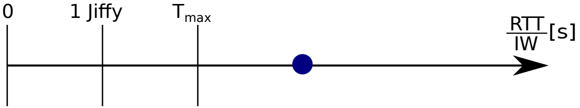
Our proposal:



- Case 1 Jiffy $< \frac{RTT}{IW} < T_{max}$
  Initial Spreading is used with $T_{spreading} = \frac{RTT}{IW}$

Proposal

Our proposal:



- Case $T_{max} < \frac{RTT}{IW}$
  Initial Spreading is used with $T_{spreading} = T_{max}$

Proposal

Our proposal:

if $\left(\frac{RTT}{IW} < 1 Jiffy\right)$

Do not use Initial Spreading

else

$$T_{spreading} = min\left(\frac{RTT}{IW}, T_{max}\right)$$

Where T_{max} is a parameter to set.

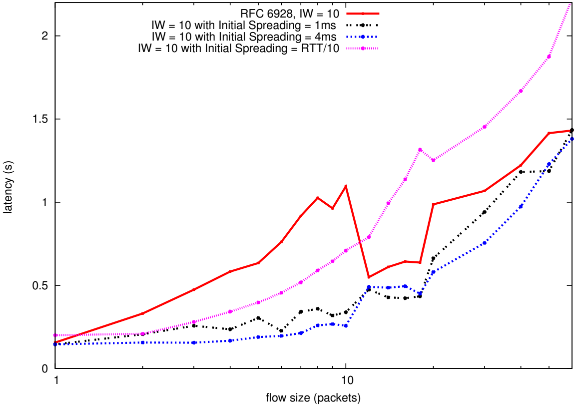
Proposal

We recommend to use T_{max} = 4 ms:

- IS works perfectly when bottleneck throughput > 4Mb/s in congested and uncongested environments
- For lower values, similar performance than RFC 6928
- Takes into account that recent kernels use a Jiffy interval of 4 ms

Experimentation

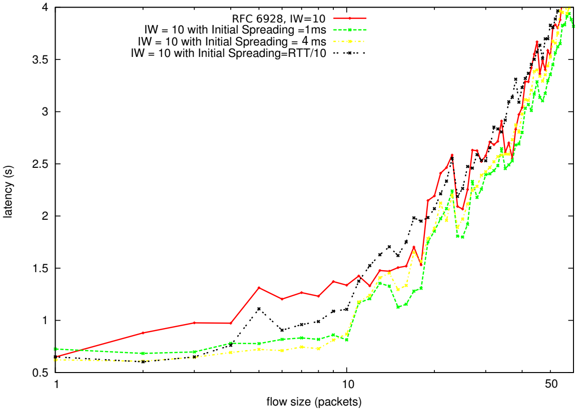
1. short RTT (> 80ms)
2. long RTT (> 500ms)



different spreadings for a delay of 40ms

Experimentation

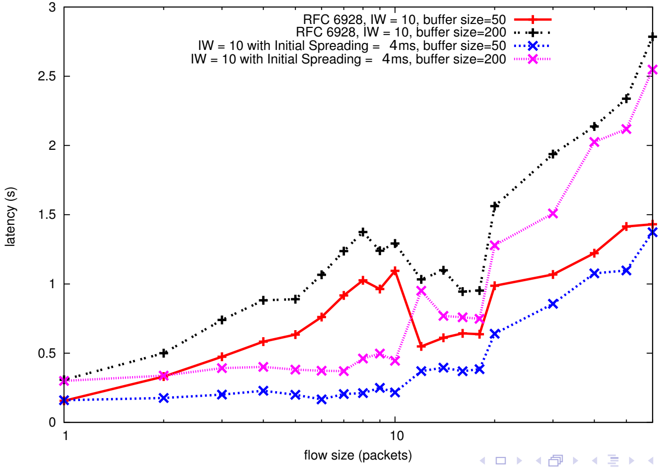
1. short RTT (> 80ms)
2. long RTT (> 500ms)



different spreadings for a delay of 250ms

Reduce the buffer size and then the Bufferbloat

- end-to-end delay: 40ms,bottleneck router: Qdisc: Pfifo Fast, bitrate: 10Mb/s
  - buffer size = 50 segments => RTT_{mean} : 150ms
  - buffer size = 200 segments => RTT_{mean} : 340ms



Implementation

- Patch available on request (linux-3.10.5)
  - 335 lines
  - Several supported options:
    - Variable Spreading
    - Bounded Spreading

- Implementation Issue:
  TSO/GSO has to be deactivated

Conclusion

Initial Spreading allows to safely enlarge the IW from 3 to 10

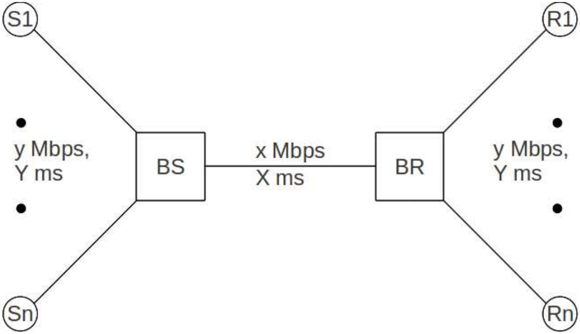
- Initial Spreading offers a simple mechanism:
  - To speed up short lived connections
  - To reduce buffer size and then Buffer bloat
  - To provide great performance enhancement for LFN

Questions

Questions ?

Testbed

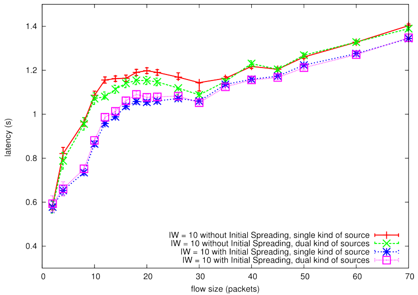
- NS2 simulations & real experimentations
- Several hundreds of iterations
- Confidence interval of 95% for each point



Network topology used

Fairness and friendliness

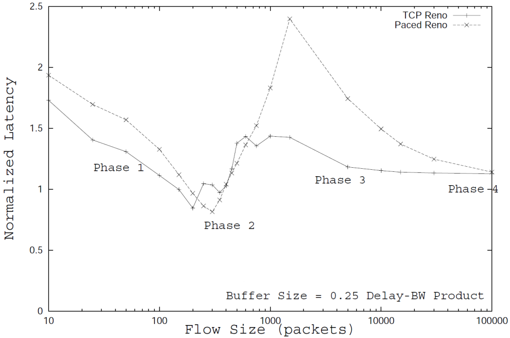
Unlike Pacing, IS performance are not mitigated by other flows



different sources sharing a bottleneck

Pacing: Flaws

- delays the congestion until a potential network collapse
- Flows synchronization



Normalized latency for flows initiated from 15 nodes

Source: A. Aggarwal, S. Savage, and T. Anderson, Understanding the performance of TCP Pacing, INFOCOM 2000

Long-lived connections

- Similar results with and without IS:
  - No flows synchronization
  - No network collapse