QUIC and SATCOM

Why do we work on QUIC over SATCOM?

Ludovic Thomas, Emmanuel Dubois, Nicolas Kuhn (CNES)
Emmanuel Lochin (ISAE)
Why do we work on QUIC performance over SATCOM?

1- Because it is already here

https://doi.org/10.1007/978-3-319-76481-8_19
https://quic.netray.io/stats.html
Why do we work on QUIC performance over SATCOM?

2- Because we (we = SATCOM) can not « split » QUIC traffic

TCP-split:
- Each TCP connection is split in 3 independent TCP connections

With TCP-split (PEP), we even do better than with IW (Initial congestion Window) of 60
SWOT analysis of QUIC and SATCOM

Strength

• 0 RTT handshake (complicated when TCP is splitted)

Weakness

• Can not be accelerated

Opportunity

• Quick deployment of new CC versions
• No PEP = cheaper ground segments

Threat

• Complexed adequate QoS for different applications (using e.g. DPI)
• Potential impact on the end-user QoE
Question

Is QUIC doing better than splitted-TCP for a SATCOM public access?
QUIC and SATCOM

Testbed description

Ludovic Thomas, Emmanuel Dubois, Nicolas Kuhn (CNES)
Emmanuel Lochin (ISAE)
Testbed description

How can we trust our QUIC experiments?

- By using Google servers

**Target A** (1 object, 5.3MB)  
**Target B** (3 objects, 11 kB)
**Testbed description**

How can we assess actual end user perception?

- By using a public SATCOM access
  - We « just » rent a SATCOM public access and connect our laptop to it
    - we get to have real end user experience
    - we have no views on the specific TCP implementations (and other operator tunings)
Testbed description

How can we assess actual end user perception?

- Beware the optimized TCP in public SATCOM access

Unknown TCP CC (adapted and specific AIMD probably based on New Reno)

BBR?
Testbed description

How can fairly compare QUIC/UDP vs optimized-TCP?

- QUIC is end-to-end and TCP is split and optimized for SATCOM
- By using one browser with the QUIC enabled option
  - (using different browser can result in comparing apples and oranges)
Testbed description

How can we see if we are the only concerned by this issue?

- By making our code available
- https://forge.net4sat.org/cnes/quxa-public
Testbed description

What do we exactly do?

- Test unit: three web pages downloads before purging the browser profile.
- Each download: the client fetches one of the web pages and then closes the browser when the page is retrieved.
- Elapsed time between two loads is uniformly distributed between 5 and 15 seconds.

- We use Selenium automation tools to control the browser and retrieve W3C metrics
QUIC and SATCOM

Results

Ludovic Thomas, Emmanuel Dubois, Nicolas Kuhn (CNES)
Emmanuel Lochin (ISAE)
**QUIC and SATCOM**

**Large page download (target A)**

Same PLT for both cases:

- First load is done with TCP
Once QUIC is know to be possible
- PLT is doubled with QUIC
- Strange increase in noQUIC PLT (probably some ISP tuning)
Focus on the load 2

Test set: 2018-08-29
Plotted: 40/40 test units

CDF du PLT pour le load 2
Sequence number (target A)

- Stable and high throughput with splitted-TCP
- It takes a while for QUIC to get out of the slow start and getting up to speed
QUIC and SATCOM

Time to responseStart (target A)

Test set: 2018-08-29
Plotted: 40/40 test units
Elapsed time between connectionStart and responseStart

- **ChromeNoQuic**
- **ChromeQuic**

<table>
<thead>
<tr>
<th>Load</th>
<th>Elapsed time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load 1</td>
<td>p = 7.2e-01</td>
</tr>
<tr>
<td>Load 2</td>
<td>p = 2.5e-01</td>
</tr>
<tr>
<td>Load 3</td>
<td>p = 1.5e-08</td>
</tr>
</tbody>
</table>

CDF du TTR pour le load 2

Effectif cumulé

Test set: 2018-08-29
Plotted: 40/40 test units
QUIC and SATCOM

Page Load Time and Time to responseStart (target B)

Test set: 2018-08-03
Plotted: 40/40 test units
Elapsed time between connectionStart and contentLoaded

Test set: 2018-08-03
Plotted: 40/40 test units
Elapsed time between connectionStart and responseStart
QUIC and SATCOM

Conclusion

Ludovic Thomas, Emmanuel Dubois, Nicolas Kuhn (CNES)
Emmanuel Lochin (ISAE)
Is QUIC doing better than splitted-TCP for a SATCOM public access?

- For small files, QUIC wins
- First data bytes arrive earlier
- For large files, splitted-TCP wins
- Issue is « getting up to speed »
- More info on the paper:
What is next?

• **Short term solution:**
  • Send relevant informations to the QUIC server
    • IW, CC parameters, etc.

• **Long term solution:**
  • Let the server know we are on a SATCOM access
  • Work on a specific QUIC CC for SATCOM access
  • Waiting for the IETF-QUIC release
QUIC and SATCOM

Couple words on the tools

Ludovic Thomas, Emmanuel Dubois, Nicolas Kuhn (CNES)
Emmanuel Lochin (ISAE)
QUIC and SATCOM

**Few words on the transport experiment automation**

Setting up such experiments takes a while

Running transport level experiments is costly

- OpenBACH: [http://www.openbach.org](http://www.openbach.org)
- Experimental work – we would be happy to have feedbacks
- Open-source orchestration tool
- Based on simple unit jobs (e.g. tcp_probe, set a VoIP server, set a HTTP server, etc.)
- Based on many open-sources project
Few words on the transport experiment automation

Example of what can be done with OpenBACH

- Network metrology (QoS metrics)
- Multipath transport scenario
- (ongoing) Integrated QUIC related scenarios
  - To see if the same trend (BBR and CUBIC fairness) could be seen in SATCOM
  - To automatize TCP fairness experiments with an open-source tool