Delay-based Metric Extension for the Babel Routing Protocol

draft-ietf-babel-rtt-extension-00

Baptiste Jonglez, Juliusz Chroboczek

July 24, 2019 — IETF 105, Montreal
Plan

Quick overview of the "delay-based metric" extension

Implementation status

Updates to the draft
Overview of the extension

Main use-case: **overlay networks**. The routing protocol has no idea of the underlying network topology.

**From Marseille to Lille: through Paris or through Tokyo?**

Figure: Overlay network: red links are tunnels.
Main idea

Measure RTT on each link and derive a metric from it.

Difficulties

We want to reuse Babel messages, and Babel is asynchronous (no ping-like measurement possible)

Solution: Mills’ algorithm, used in NTP.
RTT measurements: Mills’ algorithm

Figure: $\text{RTT}_{A \rightarrow B} = \Delta - \Delta' = (t_2 - t_1) - (t'_2 - t'_1)$
RTT measurements: Babel messages

Figure: Timestamps are transported as sub-TLVs in Babel messages
### Sub-TLV format

**Hello:**

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------+---------------------------------+--------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type = 3</td>
<td>Length = 4</td>
<td>Transmit timestamp ( t_1 )</td>
<td></td>
</tr>
<tr>
<td>+-----------+---------------------------------+--------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transmit timestamp (continued)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------+---------------------------------+--------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**IHU:**

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------+---------------------------------+--------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type = 3</td>
<td>Length = 8</td>
<td>Origin timestamp ( t_1 )</td>
<td></td>
</tr>
<tr>
<td>+-----------+---------------------------------+--------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Origin timestamp (continued)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------+---------------------------------+--------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Receive timestamp ( t'_1 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------+---------------------------------+--------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Receive timestamp (continued)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------+---------------------------------+--------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
From RTT samples to route selection

1. RTT samples
2. Smoothing
3. Smoothed RTT
4. Mapping
5. Penalty
6. Metric
7. Hysteresis
8. Link cost
9. Route selection
Figure: Example of metric computation based on smoothed RTT (from reference implementation)
More details

- See Juliusz’ presentation at IETF 104
- Full research report here:
  https://hal.inria.fr/hal-00954373
## Implementation status

### Implementations

- **babeld**: implemented by myself since version 1.5.0 (May 2014)
- **Bird**: not implemented yet, but Toke expressed interest
- **Other implementations** (Quagga/FRR, Pybabel, Sbabeld): no expressed interest that I know about

### Usages in the wild

- **Nexedi**: worldwide overlay network. Used in production for years.
- **Althea**: mix of P2P wireless links and long distance internet links. Used in production.
- Evaluation for inclusion in **LibreMesh**, used in several Community Networks.
Editorial updates to the draft

- Clarify how timestamps work with unicast Hello
- Typo: granularity of timestamps is 1 $\mu s$, not 1 $ms$
- Clarify that timestamps are unaligned 32-bit values (there are no empty fields in the sub-TLVs)
- Update references to RFC6126bis once it is published
### Discussion: updates to the draft?

#### RTT measurement loophole

- Currently, we need Hello and IHU in the same packet to compute a RTT. No requirement for this in Babel although it makes sense.

- **Solution 1**: add transmit timestamp in IHU messages?
  - **Caveat**: large overhead (4 bytes per neighbour)
  - Two possibilities:
    - new sub-TLV type: breaks compatibility (flag day)
    - append the timestamp to the current sub-TLV format: old implementations will ignore it

- **Solution 2**: new "Timestamp" TLV? Less overhead, but breaks compatibility

- **Alternative solution**: just specify that "a Hello SHOULD always be sent alongside IHUs"