Transport Layer Caching
Mechanisms and Optimization
ICCRG meeting @ IETF-85, Atlanta, GA, USA

Nestor Tiglao
INESC-ID/IST
nestor.tiglao@inesc-id.pt
Outline

- Background
- Transport layer caching model
- Main results
- Related IETF work
- Congestion Control
- Summary
Poor TCP Performance in Wireless Multi-hop Networks

- Higher bit error rates and packet loss
- Underlying MAC protocols (exponential back-off, hidden/exposed nodes)
- TCP end-to-end error and congestion control mechanisms

Reference: [3]
Proposed Solutions

- **Distributed TCP Caching (Dunkels et al., 2004)**

```
Sender | A | B | C | Receiver
------|---|---|---|---------
TCP Segment |   |   |   | TCP ACK

Segment Lost
CCTRL: Retransmit Segment
```

- **TCP Support for Sensor Networks (Braun et al., 2007)**

```
Sender | A | B | C | Receiver
------|---|---|---|---------
TCP Segment
TCP RTO
Retransmit Segment

TCP Segment is acknowledged... Regenerate ACK
TCP ACK
```

Reference: [3]
Transport Layer Caching Model

- Analyze benefit of intermediate caching in lossy networks, e.g., wireless sensor networks
- Probabilistic model
  - Link-layer component
  - Transport-layer component
- Define: Probability of Effective Progress of the packet, $PEP_i(t, h)$
Analytical Results

- Reliable delivery
- Based on DTSN semantic – DATA, NACK, MACK (MAC –layer ACK)
- RTS/CTS disabled
- Performance metric: Transmission cost, $C_i$
- Function of cache size, $C_n$
- Effective error probability associated with the transmission of a packet taking into account the maximum number of MAC retries, $\pi$
Analytical Results
Single flow: Cost vs. Cache Size
Analytical Results
Single flow: Cost vs. Hop Length

![Graph showing cost vs. hop length for different values of Cn. The graph has a logarithmic scale on the x-axis and a linear scale on the y-axis. Legend includes points for Cn = 0, Cn = 1, Cn = 5, Cn = 10, and Cn = 20.]
Analytical Results
Single flow: Cost vs. Retry Limit

![Graph showing the relationship between cost ($C_1$) and retry limit ($r$) for different retry limit values ($C_n$). The graph demonstrates that as the retry limit increases, the cost initially decreases and then stabilizes or increases slightly.]
Analytical Results

Concurrent flows: Cache Partitioning

- Caching weight assigned to flow $i$ at node $n$, $\omega_i^n$
- At each node, fraction of cache memory allocated to each flow $i$ that crosses the node $n$

\[ \rho_i^n = \frac{\omega_i^n}{\sum_{j=1}^{F_n} \omega_j^n} \]

- Cache Partitioning Policies
  - Uniform
  - Increasing
  - Decreasing
  - U Shape
  - Inverted U Shape
Cache Partitioning Example

Raw weights

Red flow – Uniform
Blue flow – Decreasing
Green flow – U Shape

Normalized weights
Concurrent Flows
Grid Topology
Concurrent flows: 10x10 Grid Network
Uniform Error Rates

\[ C_n = 20 \]
\[ l_i = 9 \]
Concurrent Flows
Non-uniform Error Rates

- Heuristic: assign higher weight to flows experiencing higher error rates and with greater hop length

\[
\omega_i^{\phi_i(k)} = \left( \pi^{DATA}_{\phi_i(k), \phi_i(k+1)} \cdot \pi^{NACK}_{\phi_i(k+1), \phi_i(k)} \right)^\alpha (l_i)^\beta
\]
Concurrent Flows: 10x10 Square Network
Non-uniform Error Rates

vertical $\pi = 0.2$
$C_n = 20$
$l_i = 9$
Concurrent Flows: 9x5 Rectangular Non-uniform Error Rates

vertical $\pi = 0.2$

$C_n = 20$

$I_i = 9$
Concurrent Flows: 10x10 Grid
Convergecast Pattern

1 sink node
99 source nodes
\( C_n = 20 \)
Simulation Results

- ns-2 simulator
- Performance evaluation
  - Better energy efficiency
  - Improve goodput
  - Better delay performance
  - Can improve fairness (Jain’s fairness index)
- Refer to [7]
TCP options
  - Cachable segments are supplied with a “Content Label”
  - TCP ACKs contain “Content Request”

Cache can send segments on behalf of the sender

Perhaps cache partitioning can be explored
  - Example: CDNs
Congestion Control

- Window-based/Rate-based
  - AIMD approach

- Bandwidth Delay Product (BDP-based)
  - BDPs represent the maximum amount of unacknowledged data that are allowed in flight at any moment
  - Multi-hop wireless networks based on spatial reuse, e.g. 802.11 MAC, BDP = 1/4 x path length
Congestion Control
Dependency on BDP

**Network Type: Long Fat Pipe**
Congestion Control Algorithm:

1. increase transmission window via adding an option into TCP header
2. develop more efficient AIMD-style congestion control algorithms

**Network Type: Medium Sized Pipe**
Congestion Control Algorithm:

1. TCP
2. max transmission window is 64KB
3. AIMD congestion control algorithm

**Network Type: Small Sized Pipe**
Congestion Control Algorithm:

1. costly AIMD-style algorithm
2. need new strategies

BDP of an $n$-Hop Linear Chain

<table>
<thead>
<tr>
<th>Number of Hops</th>
<th>BDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 2, 3</td>
<td>1</td>
</tr>
<tr>
<td>4, 5, 6</td>
<td>2</td>
</tr>
<tr>
<td>7, 8, 9</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
</tr>
</tbody>
</table>

Reference: [6]
Summary

- Transport layer caching can significantly improve performance in loss recovery
- Cache partitioning is crucial
- New approaches to congestion control
- Possible applications
  - Enhancements to CA-TCP
  - Caching-enabled transport protocol for LLNs
Thank you!
References


