Performance Analysis of the HTTP Protocol on Geostationary Satellite Links

Hans Kruse
Ohio University
Mark Allman
NASA LeRC/Sterling Software
Jim Griner, Diepchi Tran
NASA LeRC

Overview

- Network Reference Points
- The HTTP 1.0 and 1.1 Mechanisms
- Experimental Setup
- TCP and HTTP Configuration
- Results and Future Work
Why HTTP

- The Obvious Answer:
  "Millions of Web Browsers…"
- The not-so-obvious Answer:
  - HTTP is a very generic multi-file transfer protocol with content/encoding awareness
  - Very well optimized HTTP servers are available
  - HTTP contains intrinsic proxy support mechanisms that allow regional caching of data

Network Reference Points

We are here

```
End Users
```

```
"DirecPC"
```

```
"Corporate VSAT"
```

```
"ISP"
```

Reference Points cont...

- Interface “a”
  - Very small number of users
  - Traffic is bursty, user wants good response time, protocols dominate performance
- Interfaces “b” and “c”
  - Large and varying number of users
  - Traffic is more random, performance depends on protocols and congestion control; fairness is desirable

The HTTP 1.0 Mechanism

```
[Diagram of HTTP 1.0 mechanism]
```
The HTTP 1.1 Mechanism

The Experimental Setup
TCP Configuration

• Standard BSD "reno" stack
• Large window support (RFC 1323)
  – experiment uses 8, 16, 64, and 96Kbytes
• Bug fixes in the NetBSD stack
  – Initial window starts with one segment
  – Acknowledgments are generated according to the standard

HTTP Configuration

• Apache Server (HTTP 1.0 and 1.1)
  – Persistent connections in HTTP 1.0
• Netscape browser
• Netscape allows multiple connections
  – experiment uses 1, 4, 8, and 16
• Experimental HTTP 1.1 client
• Increased initial TCP window support
Comparing HTTP 1.0 and 1.1
Data Flow Comparison

The Larger TCP Initial Window

Modified Initial Window


Page 13

Page 14
What settings are important?

Comparing 1.0 multi-connection and 1.1 pipeline

Modeling Slowstart

- Slowstart creates an exponential increase in the data flow, up to the channel bandwidth.
- Delayed acknowledgements change the rate of increase.
- HTTP 1.0 requires a little extra work, results for HTTP 1.1 are shown here.
Are there unknown effects?

Experiment vs. Slow Start Model

TCP Window (KBytes)

Response Time (sec)

LeRC
LeRC Exp.
oufr
oufr Exp.
acts
acts Exp.

Maybe a few ...

Experiment vs. Model - Modified Initial Window

TCP Window (KBytes)

Response Time (sec)

acts
acts Exp.
Test
Test Exp.
Implication for the Service Provider

<table>
<thead>
<tr>
<th>Page</th>
<th>Best Time (sec)</th>
<th>Size (Kbytes)</th>
<th>Rate (KB/Sec)</th>
<th>Utilization</th>
<th>No. of Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>/acts</td>
<td>3.79</td>
<td>100</td>
<td>26.41</td>
<td>14%</td>
<td>7.1</td>
</tr>
<tr>
<td>/LeRC</td>
<td>3.00</td>
<td>49</td>
<td>16.36</td>
<td>9%</td>
<td>11.5</td>
</tr>
<tr>
<td>/oufr</td>
<td>6.89</td>
<td>491</td>
<td>71.23</td>
<td>38%</td>
<td>2.6</td>
</tr>
<tr>
<td>/Test</td>
<td>2.99</td>
<td>29</td>
<td>9.70</td>
<td>5%</td>
<td>19.3</td>
</tr>
</tbody>
</table>

Desirable Configuration:

Based on T1 (1.536Mbps) Service

Conclusions and Future Work

• HTTP 1.1 pipelining outperforms HTTP 1.0.
• Performance of HTTP 1.1 can be readily modeled.
• Pipelining will create new application level problems.
• Examine the reference points “b” and “c” by introducing competing background traffic with the TCP flow under study.