Protocol Enhancement on Challenged Networks

Presented by: Gei-Tai Lin

September 14, 2004

Introduction

Required Readings

- Performance Optimizations for Wireless Wide-Area Networks: Comparative Study and Experimental Evaluation
- Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations

Performance Optimizations for Wireless Wide-Area Networks: Comparative Study and Experimental Evaluation

- Key Contributions:
  - present first detailed evaluation of application performance over commercial WWAN environments
  - implement a wide selection of optimization techniques at different layers
  - present an experiment methodology based on virtual web hosting for performing reproducible experiments over WWAN environments

File Transfer vs Web Browsing

- The throughput achieved in file transfer experiments were significantly higher than the web downloads.
- For example in similar network conditions the web download throughput for amazon.com with a total content size of 91.9 KB was 9.6 Kbps, while download of a single 50 or 100 KB file was around 30 Kbps!

TCP connections per website

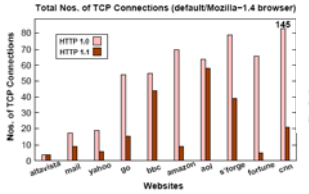

Figure 2: Total number of TCP connections opened by the client to download the main webpages of the popular web-sites with HTTP/1.0 and HTTP/1.1 protocols. Some of the web servers send 'pre-emptive FINs' and hence benefits of HTTP/1.1 are not always realized.

File Size (KB)	FTP-throughput (Kbps)
1	13.2 (1.5)
5	18.1 (0.9)
10	18.8 (2.1)
50	29.7 (3.3)
100	30.5 (3.2)

Table 2: Data throughputs achieved for ftp-downloads over WWAN wireless links using a single TCP connection. TCP achieves good throughput for larger files.

Experiment Methodology

- Virtual Web Hosting – replicated the contents of popular websites into a set of local web servers in lab with public domain names. (did anyone think that someone might have been accessing their websites and possibly causing errors in their actual experiment data?)

WWAN Experimental Testbed

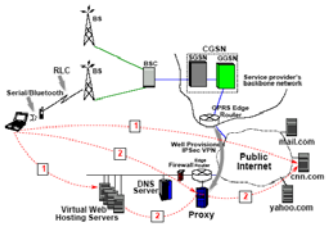

Figure 1: WWAN Experimental Testbed. In our experiments, we placed the proxy in our laboratory and then use a well provisioned IPSec VPN to 'back haul' the traffic from the cellular provider's network. The mobile client connects to the web servers through this proxy (shown as Label 2).

Differences

- Server-side Load
  - real web servers have higher load and variable load
- Pre-emptive FINs
  - some webservers close TCP connections pre-emptively to prevent outstanding connections and DoS attacks
- Web content
  - static content on virtual host, some dynamic content on real websites

Benchmarking Performance

- Used Mozilla browser version 1.4 (should they have used other browsers as well? Since all tests are automated, was there really a reason not to have tested using other browsers?)
- Mobile host kept stationary, resulting bit error rate 0-4%

Distribution of Latencies

A contributing factor of this under utilization is distribution of object sizes and due to the stop-and-go behavior of HTTP in default mode.

Client makes a GET request, receives it completely, then makes the next GET request. This results in under performance due to high WWAN link latency.

Performance Optimizations

- Application Level
  - Dynamic Content Compression
    How well it works depends on number of objects and size
  - Optimizing Using Pipelining
    (HTTP/1.0, new TCP connection for every object downloaded)
    (HTTP/1.1-default 2 TCP connections, but still suffers from significant under-utilization of the WWAN link)
    An experimental option in the standard can issue new GET requests without waiting for the entire response.
    Issues simultaneous requests and ensures full TCP utilization.
    35-36% benefit

Performance Optimizations (cont…)

- Extended Caching and Delta Encoding
  - Extended Caching both index objects by their SHA-1 fingerprint, or CHK (Content Hash Key) using CHK an identical item is never downloaded twice
  - Delta Encoding send only difference between new and old version of document Together improves web-browsing by 3-6%

Performance Optimizations

- Session-layer Techniques
  - Varying number of TCP connections

  An empirical configuration based on the network

URL-rewriting/DNS-Rewriting

- client makes one DNS look up. works like web caching in that the server serves the stuff to you and looks it up

Performance Optimizations (cont…)

- Transport-layer Techniques
  - TCP WWAN

  avoid slow start and use aggressive recovery
  - Custom Transport Protocol

    * optimized specifically for GPRS networks

  GPRS link layers offer reliable in-order data delivery so UDP-GPRS can make many more assumptions about the underlying network

    * no need for congestion avoidance, already implemented mechanism for shared bandwidth on GPRS - improve download web 7-14%

Performance Optimizations (cont…)

- Link Layer Techniques alter FEC rates and ARQ

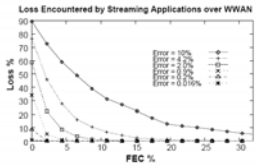

Figure 7: The impact of link-layer FECs on the data loss rate experienced by interactive streaming applications over WWANs. (WWAN trace-driven simulations).

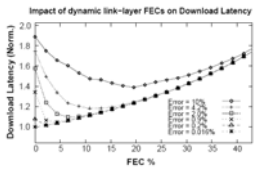

Figure 6: Impact of FEC-based link-layer data recovery for WWAN links on bulk data downloads under different channel conditions. (WWAN trace-driven simulations).

Results

Website	No-reconf.-I		No-reconf.-II		Client-reconf.	
	Lat.	Improv.	Lat.	Improv.	Lat.	Improv.
mail	15.7	**54.4%**	13.2	**61.6%**	12.4	**64.0%**
yahoo	11.6	**66.9%**	11.4	**67.2%**	9.9	**71.7%**
amazon	30.8	**59.6%**	27.4	**64.1%**	24.3	**68.1%**
cnn	96.2	**50.9%**	65.1	**66.8%**	59.3	**69.7%**

Table 9: Relative improvement, with respect to HTTP/1.1-default, provided by the No-reconf. and Client-reconf. based schemes. Download latency in seconds.

Non-reconf schemes include Full Compression, HTTP/1.1-Opt, DNS-Rewriting/URL-rewriting, TCP-wan, and dynamic FECS

Conclusions

- Severe Mismatch between TCP and HTTP
- Applications and Session Layers Dominate Benefits
- Use of Proxy Beneficial

Performance Enhancing Proxies Intended to Mitigate Degradations

- - link design choices can have a significant influence on the performance and efficiency of the Internet (eg high latency)
- - should be used only in specific environments and circumstances where end-to-end mechanisms providing similar performance enhancements are not available

Types of Performance Enhancing Proxies

- Transport Layer
- Application Layer

Transport Layer PEPs

- - operate at the transport layer
- - most interact with TCP, and are so called TCP PEP
- - eg, on a network where ACKs bunch together causing segment bursts, TCP PEP may be used to modify the ACK spacing to improve performance.
- - sometimes TCP PEP is called "spoofing" because it will intercept a TCP connection in the middle and terminate the connection as if the interceptor is the intended destination, but this isn't a characteristic of all TCP PEPs

Application Layer PEPs

- - operate above the transport layer
- - some regular proxies are Web caches, Mail Transfer Agents (MTA). Such proxies try to improve performance in ways that are applicable in any environment but not necessarily specific to certain link characteristics
- - Application layer PEPs on the other hand can improve application protocol and transport layer performance depending on the link type. Using PEPs as intermediate note, unnecessary overhead from application protocol like extraneous RT, verbose headers or inefficient header encoding, impacting performance on long delay and slow links.

Distribution

- - distributed or integrated?
- - integrated: PEP implemented in a single node where perf enchancement is applied.
- - eg, a single PEP component might be implemented to provide impedance matching at the point where wired and wireless links meet.
- (The use of electric circuits, transmission lines, and other devices to make the impedance of a load equal to the internal impedance of the source of power, thereby making possible the most efficient transfer of power.)
- - A distributed PEP implementation is generally used to surround a particular link for which performance enhancement is desired. For example, a PEP implementation for a satellite connection may be distributed between two PEPs located at each end of the satellite link.

Symmetry and Assymetry

- Symmetric PEPs use identical behavior in both directions, i.e., the actions taken by the PEP occur independent from which interface a packet is received.
- Asymmetric PEPs operate differently in each direction. The direction can be defined in terms of the link (e.g., from a central site to a remote site) or in terms of protocol traffic (e.g., the direction of TCP data flow, often called the TCP data channel, or the direction of TCP ACK flow, often called the TCP ACK channel).
- Whether Symmetric or asymmetric is independent of being distributed or integrated.

PEP Mechanisms

- TCP ACK Handling
- TCP ACK Spacing
- Local TCP Acknowledgements
- Local TCP Retransmissions
- TCP ACK Filtering and Reconstruction

Tunneling

- force a connection between two ends to make sure use of a certain link is used.
- A Performance Enhancing Proxy may encapsulate messages to carry the messages across a particular link or to force messages to traverse a particular path. A PEP at the other end of the encapsulation tunnel removes the tunnel wrappers before final delivery to the receiving end system. A tunnel might be used by a distributed split connection TCP implementation as the means for carrying the connection between the distributed PEPs. A tunnel might also be used to support forcing TCP connections which use asymmetric routing to go through the end points of a distributed PEP implementation.
- (Possible problem with the paper…neglects to emphasize how close PEPs need to the end systems to be effective. Example is the split connection sending, with PEP making a connection to the other end system on the first end system's behalf to make use of the scaling window)

Compression

- reduces number of bytes sent
- link compression: TCP and IP header compression are also frequently used with PEP implementations. Payload compression: increasingly important with new internet security which encrypts payload and removes the possibility of link compression because of hidden TCP and IP header information. Common compression algorithms can be applied to IP segment payloads: IF the payload is not already compressed or encrypted with TLS (security mechanism above the network layer).
- With application layer PEPs one can employ application-specific compression. Typically an application-specific (or content-specific) compression mechanism is much more efficient than any generic compression mechanism. For example, a distributed Web PEP implementation may implement more efficient binary encoding of HTTP headers, or a PEP can employ lossy compression that reduces the image quality of online-images on Web pages according to end user instructions, thus reducing the number of bytes transferred over a slow link and consequently the response time perceived by the user

Implications of Using PEPs

- Does use of PEPs break the end-to-end argument?
- Not all PEP implementations break the end-to-end semantics of connections.
- Correctly designed PEPs do not attempt to replace any application level end-to-end function, but only attempt to add performance optimizations to a subpath of the end-to-end path between the application endpoints. Doing this can be consistent with the end-to-end argument.
- Performance enhancements: eg. WWAN

Security

- In general, a user or network administrator must choose between using PEPs and using IPsec.
- If a PEP implementation is non-transparent to the users and the users trust the PEP in the middle, IPsec can be used separately between each end system and PEP. You trust the PEP, so you let it decrypt in the middle to find out how to process.
- Even when a PEP implementation does not break the end-to-end semantics of a connection, the PEP implementation may not be able to function in the presence of IPsec. For example, it is difficult to do ACK spacing if the PEP cannot reliably determine which IP packets contain ACKs of interest.

Fate Sharing

- End-to-end argument: if no state is stored in the network, then if a link fails then the connection will restore itself provided there is another path.
- PEPs there is state stored in the nodes containing the PEP, so if the node crashes all this state is lost and the connection must be terminated.

Scalability

- Increased processing power and memory requirements
- Scalability issues with respect to the use of PEPs. Placement of a PEP on a high speed link or a link which supports a large number of connections may require network topology changes beyond just inserting the PEP into the path of the traffic.
- e.g., if a PEP can only handle half of the traffic on a link, multiple PEPs may need to be used in parallel, adding complexity to the network configuration to divide the traffic between the PEPs.

Conclusions

- Technically, PEPs do not necessarily have to break the end-to-end principle.
- BUT: Fate Sharing not upheld
- Scalability conditional – is the spirit of the end-to-end principle to make the internet infinitely scalable? Network topology can be DEPENDENT on the PEP