PSM-throttling: Minimizing Energy Consumption for Bulk Data Communications in WLANs

Xiaodong Zhang
Ohio State University

in Collaborations with
Enhua Tan, Ohio State
Lei Guo, Yahoo!
Songqing Chen, George Mason University
What is Going on in Internet?

- **Wireless accesses** to Internet are very pervasive:
  - *Everywhere:* campus, offices, home, public utilities …
  - Most are supported by Wireless LANs
- **Media content** is heavily delivered in Internet:
  - Streaming: Real or Window media (> 80% in TCP)
  - HTTP-based streaming: YouTube, Google Video (TCP)
  - TCP downloading: web sites (traditional or web 2.0)
- Mobile devices are increasingly diverse and portable:
  - Laptop, PDA, iPhone, WiFi phone …

**Challenge:** mobile devices can easily exhaust their batteries by viewing streaming media.
Limited Power in Mobile Devices

Wireless Network Interface is a major power consuming source!

Question: Can we minimize WNI (Wireless Network Interface) power consumption while satisfying the QoS requirement?
802.11 Power Saving Mode (PSM) and Its Variant

- **802.11 Power Saving Mode (PSM):**
  - Wakeup every 100ms via PS poll
  - Limits:
    - Buffered packets lead to delayed TCP acknowledgements
    - Increase round trip time estimation at the sender side
    - May degrade the TCP throughput

- **PSM adaptive (PSM-A):**
  - Widely used in commercial products
  - Only sleep when no packet receives for a while (e.g. 75ms)
  - Offer less power saving, but retain TCP performance
Ideal Power Saving Condition: Sleep Well and Work Energetically

- If the media traffic can form a predictable pattern:
  - periodic bursts
  - Client WNI can sleep/work periodically
  - Transmitting: 1.346W, receiving: 0.9W, idle: 0.741W
  - Sleep: 0.048W
Forming Ideal Condition by Proxy

- Buffer and shape media traffic into periodical bursts
  - A proxy between mobile devices and media servers
  - Devices can predict packet arrival time, safely sleeping
    - minimizing power consumption without performance loss
- Limits:
  - modified client
  - Proxy needs to handle diverse communication protocols
  - Needs a dedicated and expensive infrastructure

- Reference: Chandra & Vahdat, USENIX’02
Problem Statement

- Can we minimize mobile devices power consumption while satisfying the QoS requirement, without using a dedicated proxy?
Outline

- Problem Statement
- Motivation and Overview Design
- Internet Experiments on Bandwidth Throttling
- PSM-throttling Design
- Evaluation
- Summary
Conventional TCP Throughput Control

- Works well for low volume of simultaneous requests
- **Large** volume of simultaneous requests make servers be the bottleneck
- TCP congestion control cannot well address server bottleneck problem

Full-Speed Transmission

Server

TCP Congestion Control

Internet

TCP Congestion Control

Client
Bandwidth Throttling: Serve Large Volume of Requests Simultaneously

**Bandwidth Throttling:**

Server sends packets at a limited **constant** rate (lower than end-to-end bandwidth) → More Simultaneous requests will be served
Bandwidth Throttled TCP Transmission

- Window size defaults to 256K
- ACK every two data packets
- Transfer rate throttled by the server in a constant rate
- WNI is busy receiving, no time to sleep
PSM-throttling: A Client-hinted Mechanism to Create Bursts for Power Saving

- No proxy needed
- Application independent and client-centric: only add a new function in client WNI driver
Related Technical Issues

- How can a server recognize client hints?
  - Utilizing TCP flow control mechanism: placing “hints” in receive window field in the TCP header
  - No change to servers
- An ACK packet during sleep consumes trivial power
- For a given flow burst period (T), E2E bandwidth, and average throughput (r), (win size = T × r)
  - burst and sleep times are determined
  - to sufficiently sleep, T should be larger than 2 × RTT
Contributions

- Our Internet experiments show *bandwidth throttling* is widely adopted in TCP-based media services

- **PSM-throttling**: bandwidth throttling detection and power saving at the transmission layer of the wireless client: design & prototype implementation

- Evaluation results show PSM-throttling can improve power saving by up to 75%:
  - Still can meet the QoS requirement
Bandwidth Throttling Detection in TCP-based Media Transmissions

- Major media servers, such as Windows/Real Media servers, and lighttpd server (used by YouTube), have build-in bandwidth throttling

- We confirm this by detecting server-side throttling:
  - **Choke** the transmission for 200 ms at client side: Win = 0
  - **Unchoke**: Win = Original

- If resulting traffic bursts and the TCP throughput is remained the same → server is using bandwidth throttling

- We analyze the burst patterns and throughputs of Internet traffics to verify our detection methods
Real Media Server: A Sequence without Choke/Unchoke

Packets are continuously Delivered: Tput = 350Kbps
Real Media Server: the Sequence with Periodical Choking

Bursty and keeps the TCP Throughput: $\text{Tput} = 350\text{Kbps}$
Windows Media Server: a Sequence without Choke/Unchoke
Windows Media Server: the Sequence with Periodical Choking

Bursty and keeps the TCP throughput
YouTube Server: a Sequence without choke/unchoke
YouTube Server: the Sequence with Periodical Choking

Bursts become periodical and the TCP throughput unchanged
Linux File Server: a Sequence without choke/unchoke
Linux File Server: the Sequence with Periodical Choking

TCP throughput reduced!
⇒ No throttling at the server side
PSM-throttling Design: Bandwidth Throttling Detection

- Measure the stable flow rate $r$ for a specific duration $T_0$
- Creating bursts by choking the connection for half of the time
  - Measure the flow burst rate $r'$
- If $r' \geq 2r$, server is engaging bandwidth throttling, which can be used for power saving in PSM-throttling
PSM-throttling Design: Further Increase Traffic Burst

- If the burst rate is not high enough (not fully utilizing the wireless channel):
  - Client waits for about 20 ms so that there are enough packets buffered at the AP
  - PS Poll to receive the buffered packets → More bursty!
Prototype Implementation

- Add a new function to client WNI driver:
  - D-Link DWL-G520 wireless card (Atheros chipset)
  - Madwifi 0.9.2 driver under Linux 2.6.18

- Comparing with 4 other schemes
  - CAM: Continually Aware Mode (no power saving)
  - CC: Client-Centric (IWQoS’04)
  - PSM: Power Saving Mode (IEEE standard)
  - PSM-A: PSM Adaptive (focus on comparing with)
Performance Evaluation

- Lab and Internet Experiments:

media server
Performance Evaluation Results

Fig. 10. TCP-based Windows media streaming

Fig. 11. TCP-based RealNetworks media streaming

- Windows and Real media streaming
Performance Evaluation Results (cont.)

Fig. 12. YouTube pseudo streaming

Fig. 13. HTTP downloading with bandwidth throttling

- YouTube pseudo streaming; HTTP downloading
Summary

- Internet bandwidth is no longer critical bottleneck, but servers due to increasingly high volume of simultaneous requests
  - Conventional traffic smoothing methods are no longer useful
  - Bandwidth throttling addresses this problem
  - We take this unique opportunity to design and implement PSM-throttling to create receive/sleep periods for power saving on WNI

- PSM-throttling has the merits of:
  - No additional infrastructure support
  - Application independent and client-centric
  - Save up to 75% of the power and keep the TCP throughput

- Applicable for cellular networks and WiMax networks
- Hints from applications can further improve PSM-throttling
Thank you!

Enhua Tan: etan@cse.ohio-state.edu
http://www.cse.ohio-state.edu/hpcs/
PSM-throttling: Adaptation to Server Transmission Rate

- Dynamically adjust the receive window size
- If the throughput is not decreasing, increase the recv wnd size
- If sleep time falls below a RTT, decrease the recv wnd size
Evaluation of PSM-throttling Adaptation to Server Transmission Fluctuations