Scalable Data Transfer Applications

<table>
<thead>
<tr>
<th>End-system / Application-level</th>
<th># of Receivers</th>
<th># of Senders</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One</td>
<td>One</td>
</tr>
<tr>
<td>ftp, traditional apps</td>
<td>Many</td>
<td>Many</td>
</tr>
<tr>
<td>web downloads</td>
<td></td>
<td></td>
</tr>
<tr>
<td>video-on-demand</td>
<td></td>
<td></td>
</tr>
<tr>
<td>server push</td>
<td></td>
<td></td>
</tr>
<tr>
<td>chat rooms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>video conferencing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>multiplayer games</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Scalable Wide-area Upload

[Bistro00]

Bill Cheng

http://merlot.usc.edu/cs530-s10
Scalable Data Transfer Applications

End-system / Application-level

<table>
<thead>
<tr>
<th># of Receivers</th>
<th># of Senders</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>One</td>
<td>One</td>
<td>ftp traditional apps, web downloads, software distribution, video-on-demand, server push, chat rooms, video conferencing, multiplayer games</td>
</tr>
<tr>
<td>Many</td>
<td>Many</td>
<td>Bistro!!</td>
</tr>
</tbody>
</table>

To the best of our knowledge, there is no existing work on making many-to-one communication at the application layer scalable and efficient.

Who Is Working on Uploads?

What Are Upload Applications?

- Hard deadlines
  - IRS income tax submission
  - paper submission
  - real-life events

- No hard deadlines
  - Internet-based storage
  - Data warehousing

Why is Upload Different?

- many-to-one data transfer
- read vs. write
  - traditional solution such as replication of data (caching), replacement of data, etc. won't help
  - fault tolerance, security
- contention for service rather than data
- data consumed later (will exploit this)
- replication of services and resources for a single event is expensive, inflexible, & not scalable

Traditional Approaches

(at the application layer)

- Increase capacity
- Spread the load ... over time, space, or both
- Change the workload

Examples

- data replication
  - ftp mirroring, web caching
- data replacement
  - multi-resolution images, video
- service replication
  - DNS lookup, NTP
- server push
  - news download, software distribution

Traditional Approaches (Cont...)

Example: Akamai

- Relieve web download hotspots through data replication (caching)
- Use their own network of servers, with strategic placement of servers around the world
  - > 2700 servers
  - > 45 countries
  - > 150 networks
- Clients include: Microsoft, Paramount, Wired, CBS Sports, Nike, BBC America, Apple, ...

Why are there hotspots?

- real-life events
- availability of new data
Our Goals

- A single infrastructure (termed Bistro) for all data collection needs
  - good performance (for both service providers and users)
  - scalable (shares resources among all service providers)
  - secure (one service provider does not have to trust another)

Current State of Affairs for Uploading

- Independent data transfers over the Internet, i.e., TCP/IP
  - TCP/IP shares bandwidth fairly
  - individual clients experience poor performance when number of clients is large (if transfer time is long enough to see other connections)
  - TCP/IP is here to stay

Not scalable!

Key Observations

- Existence of hot spots in uploads is largely due to approaching deadlines
- Exacerbated by long transfer times
- Problem: too much data ... too little time ...

Key Observations (Cont...)

- What is actually needed is an assurance that specific data was submitted before a specific time
  - i.e., we need a commitment of what and when a submission took place
  - Then the transfer of that data needs to be done in a timely manner, but does not have to occur by the deadline
  - unlink downloads, the data may not be consumed at the server right away
  - if a piece of data arrives after the deadline, we just need to guarantee that it’s exactly the same piece of data that was committed before the deadline

Solution with Bistro

- Before deadline:
- Traffic at/near Destination Server:

A Solution to Upload with Deadlines

- A Bistro is like an e-Post Office, built to handle certified e-submissions
- A bistro can be installed on an IRS server or a tax partner’s server
- A bistro is installed on an IRS server or a tax partner’s server
- Note:
  - Multiple events may be going on concurrently or overlapping, each with a different destination server
A Solution to Upload with Deadlines

Step 1: Real-time fingerprinting & timestamp
- A client generates a fingerprint for the document (tax return)
- Destination Bistro issues a timestamped and certified e-ticket

Step 2: Low-latency upload to any intermediary (commit) (client-push)
- A client verifies the digital signature on the e-ticket, encrypts the document, and uploads the encrypted document to any bistro (or a designated bistro for a tax partner)

Step 3: Timely transfer to final destination (large scale data transfer) (server pull)

Who is Trusted with What?
- Event Operator (IRS) trusts the Destination Bistro for this event
- End User (tax payer) trusts its Client software and the Destination Bistro for this event
- Destination Bistro trusts the Bistro software to generate a pair of public and private keys \( (K_{pub}, K_{priv}) \) for this event

Bistro FAQ
- Why do you need step (2)? Why can’t the destination server get the document directly from a client in step (3)?
  - A client can be behind a firewall or a client’s machine can be turned off.
  - A bistro is always on the public Internet, and may be subject to attacks. Therefore, all documents on a bistro must be encrypted.
- Why did you show that step (2) is done before the deadline?
  - Step (2) is the commit step, it does not need to be done before the deadline since the only transaction that is required to be completed before the deadline in step (1). However, to complete a client’s transaction (so that the client can leave or shut down its PC), we must push the encrypted data out of the client’s PC.
  - Since there can be many bistros, this will not cause a traffic jam. Also, most of the data transfers during this step are localized.
**Bistro FAQ (Cont...)**

- **Can a fingerprint be forged?**
  SHA1 is the state-of-the-art electronic fingerprinting algorithm. It generates a 160-bit fingerprint for an any-size document. If you modify a single bit in a document, the new document has a completely different fingerprint. There is no known algorithm that can forge a SHA1 fingerprint while maintaining the integrity of a document. (The Bistro system is not tied to a particular fingerprinting algorithm. Please see below.)

- **Can the destination server be under denial-of-service attack?**
  Yes. That’s one weakness of the Internet. However, you can setup mirrors for the destination server by copying the credentials of the destination server onto alternative servers. Nevertheless, in the current Bistro system, this needs to be done ahead of time.

- **How secure is the encryption? Can it be cracked?**
  The strength of encryption is usually a function of the algorithm and key size. The Bistro system is not tied to a particular algorithm or key size. It lets the event operator choose these at the time an event is setup. As new and more secure algorithms become available, the system will need to be upgraded to support them.

**Opportunities to Speed up Data Transfers**

- **X & Y send simultaneously to D -- 2 units of time**
- **X & Y send simultaneously to Z then to D -- 3 units of time**
- **??? -- 1.2 units of time**

**Advantages of Bistro**

- Shares resources and a single infrastructure
- Replaces a traditionally synchronized client push solution with a non-synchronized combination of client-push and server-pull
- Eliminates hot spots by spreading most of the demand on the server over time, by making the actual data transfer independent of the deadline
- Deployable today, i.e., no change required inside the network
- Gradual deployment over a public, private, or mixed infrastructure of hosts
- More dynamic and therefore more adaptive to system and network conditions

**Vision**

- A bistro in every administrative domain e.g., co-located with web servers or mail servers
- Entire network of bistros collects data for one application/agency one day and for another application/agency the next day
- Use the Bistro infrastructure for other large scale data gathering, transfer, and storage needs
Some Research Problems

- Resource location and discovery
- Placement and assignment
- Security
- Large scale data transfer

Online Digital Signatures

Why digital signature?
- Integrity
- Authentication
- Nonrepudiation

Our Approach

- No batching scheme
  - Send \(DS[I_j]\) to each Client \(1 \leq j \leq B\)

- Simple batching scheme
  - Send \(DS[I_j] = D + DS[I_1 + \cdots + I_j]\) to each Client \(1 \leq j \leq B\)

Real-Time Timestamp

- Using digital signature to generate real-time timestamp

- Extra information to be sent to clients

CS530
Bistro Improvements
Bill Cheng
http://merlot.usc.edu/cs530-s10
Our Approach (Cont...)

- Tree-based batching scheme

\[ R = H(H(I_1) + H(I_2) + \ldots + H(I_n)) \]

Send to \( C \)

- Batch-based schemes do reduce a server's CPU load (where hashing is not the dominant factor)

Performance Evaluation

- Batching schemes have considerable advantage but cost relatively little (where hashing is not the dominant factor)

Commit Problem

**Extreme Cases**

- Final destination is the only bistro
- All hosts are bistros
- Each organization has a local bistro (same granularity as NNTP servers, DNS servers, etc.); in this case commit problem still non-trivial if the local bistro is not part of the public Internet

Commit Problem

**Middle Ground**

- Assignment problem
  - bistros are fixed & the difficulty is in assigning clients to the bistros
  - Placement or selection (plus assignment) problem
    - location of bistros is flexible
    - choose \( M \) out of \( N \) bistros, as well as assign clients to chosen bistros

Why is this different from downloads?
Performance Study

- Simulation setup (using ns2 & GT-ITM)
  - Transit-stub graph with 152 nodes
  - 2 transit domains, with avg 4 nodes each, edge between pair of nodes with prob 0.6 & each node having 3 stub domains connected
  - Stub domains have on avg 6 nodes each, edge between pair of nodes with prob 0.2
  - Capacity of transit-transit edge is 1 Mbit/s
  - Capacity of transit-stub or stub-stub edge is 256 Kbits/s
  - 96 simultaneous uploads with files unif. distr. between 100 KBytes & 2 MBytes
  - Low background load (30%); high background load (70%)

Performance Results

Performance gains mainly due to parallelism

Performance Study

- Note: seq. uploads to single server should be approx 3000 sec, and avg. transfer time of one client should be approx 33 sec

- Note: simultaneous uploads to single server takes approx 3000 sec, but avg. transfer time of one client takes approx 2000 sec

- Performance metrics used
  - Mean transfer time over all clients
  - Total (or maximum) transfer time

- Policies
  - Random, ping-v, ping-m
  - Unrealistic heuristic (approx. lower bound)

Performance Results

Legend:
- EID: Event ID
- Xpub: Event Public Key
- Xpriv: Event Private Key
- KX: Secure X Public Key
- KX': Secure X Private Key
- T: Data to upload
- t: Message Digest
- s: Timestamp
- X: Ticket
- R: Receipt

Date Transfer

Bistro

Event Owner (ISP)  Event Owner (ISP)  Client (Taxpayer)  Destination  Binary X

Table:

- Event ID
- Event Public Key
- Event Private Key
- Secure X Public Key
- Secure X Private Key
- Data to upload
- Message Digest
- Timestamp
- Ticket
- Receipt

Drawings of graphs and diagrams related to performance study and Bistro scenario.
Large-scale Data Collection

- Destination server needs to collect data from all other bistros but how?
- Several simple approaches
  - one-by-one: poor resource utilization due to non-shared bottleneck link
  - all-at-once: longer transfer time
  - spread-in-time-GT
  - concurrent-G
  - network congestion
- application level re-routing
  - avoid congested links
  - devise a coordinated transfer schedule

Opportunities

- Simulation setup (using ns2 & GT-ITM)
  - 7 other bistros, each with a total amount of data unif. distr. between 25 MBytes & 75 MBytes and the total amount of data in bistros is 350 MBytes.

Opportunities (Cont...)

- Host D
- Host Z
- Host X 1 Unit of Data
- Host Y 1 Unit of Data
- L1/1
- L2/1
- L3/2
- N1
- N2
- N3

Scenarios:
- X & Y send simultaneously to D -- 2 units of time
- X sends to D, then Y sends to D -- 2 units of time
- X & Y send simultaneously to Z then to D -- 3 units of time
- X sends to D, Y sends to Z then to D -- 1.5 units of time
- ??? -- 1.2 units of time

Storage Space Requirements

- needs fairly little additional storage space
  - pathsync < 4%, pathmerge and pathdelay < 41%

Effect on Other Traffic

- no significant effect on throughput of other traffic
  - (< 17%)
Contributions Thus Far

- First effort to study many-to-one communication problem at the application layer & attempt at stating fundamental obstacles
- Proposed a reasonably general framework
- Proposed solutions to all parts of the problem
- Suggested some open problems

Related Work

- Akamai and other content distribution networks
- Napster
- A variety of server selection problems
- Internet security

Related Work (Cont...)

- Many-to-one communication at IP level & within Active network framework
  - Gathercast [Badrinath & Sudama 98]
  - Concast [Calvert et al. 97]
- Wide area applications
  - wide-area download applications: e.g., Akamai [Karger et al. 97]
  - Napster type systems, e.g., [Kong & Ghosal 99]
  - application layer multicast: e.g., [Chu et al. 00]
- Client-side server selection
  - statistical: e.g., [Seshnm et al. 97]
  - dynamic: e.g., [Carter & Crovella 97] [Sayal et al. 98] [Dykes et al. 00]

Related Work (Cont...)

- Application level re-routing
  - alternate paths [Savage et al. 99]
  - Detour [Savage et al. 99]
  - RON: resilient overlay network [Andersen et al. 01]
- Online batch-based digital signature schemes
  - modification on cryptographic algorithm [A. Fiat 89]
  - one-time signatures used in secret key system [Lamport 79, Merkle 88]

Vision

- A bistro in every administrative domain
  - e.g., co-located with web servers
- Entire network of bistros collects data for one application one day and for another application the next day
- Use the Bistro infrastructure for other large scale data gathering, transfer, and storage needs

Participants

- Faculty Members:
  - Leana Golubchik
  - Samir Khuller (UMD)
  - Cheng-Fu Chou (NTU)
- Research Staff:
  - William C. Cheng
- Students:
  - Yung-Chun Wan (UMD)

Contact Information

- Prof. Leana Golubchik
  - Computer Science Department
  - University of Southern California
  - Los Angeles, CA 90089
  - Email: leana@cs.usc.edu
  - Voice: (213) 740-4524
  - Fax: (213) 740-7285
  - URL: http://cs.usc.edu/~leana

Participants Contact Information

- Prof. Leana Golubchik
  - Computer Science Department
  - University of Southern California
  - Los Angeles, CA 90089
  - Email: leana@cs.usc.edu
  - Voice: (213) 740-4524
  - Fax: (213) 740-7285
  - URL: http://cs.usc.edu/~leana