P2P Tutorial

Thanks for the slides from Keith Ross and Arnaud Legout.
Definition: Overlay

- Overlay Network
  - Network at the application layer (layer 7)
Definition: Overlay

Why do we need overlays?

- Create a service that is not (or that cannot be) provided by the network (layer 3)
  - Create an application layer service

Example of services

- Application layer multicast
- Content Delivery Network (CDN)
Definition of P2P

1) Significant autonomy from central servers

2) Exploits resources at the edges of the Internet
   - bandwidth
   - storage and content
   - CPU cycles

3) Resources at edge have intermittent connectivity, being added & removed
Intuition

- P2P

- Client-server
Definition: P2P

- Overlay Network vs. P2P applications
  - A P2P application forms an overlay network
  - An overlay network is not always a P2P application
  - Trend to define a P2P application as overlay network formed by end-users
  - Depends on the definition of P2P
It’s a broad definition:

- P2P file distribution
  - BitTorrent
- P2P file sharing
  - Gnutella, KaZaA, eDonkey, etc
- P2P communication
  - Instant messaging
  - Voice-over-IP: Skype
- P2P streaming
  - Live
  - On-demand
- P2P computation
  - seti@home
- Anonymous Web browsing
  - Tor
- Distributed databases
  - DHTs: Chord, Tapestry, Kademlia, etc
The New P2P Paradigm

- Why P2P applications became popular recently
  - High speed Internet connections
  - Power shift from servers to end-users

- P2P applications are a true revolution
  - Aside TCP/IP and the Web
P2P: Worldwide Computer

Alice’s home computer:
- Matching gene sequences
- Serving video fragments for video on demand
- Backing up others' files
- Relaying voice communications
- Providing anonymity services

Moonlighting
- Payments come from:
  - biotech company
  - movie publisher
  - backup service

Challenges
- heterogeneity
- security
- incentives
- ISPs
Major results

End of 2004

- **BitTorrent is dominating the Internet traffic**
  - 30% of the internet traffic!
- Shift of demand from **music to movies**
- Major sources of torrent file discontinued due to legal actions (e.g., Suprnova.org)
Major results

2005

- Shift from BitTorrent to eDonkey

- But, this is not the end of the story
  - BitTorrent heavily used for legal contents
  - Decentralized versions of BitTorrent
  - New large BitTorrent services (ThePirateBay, mininova, isohunt, etc.)
More Recent Results

- Unofficial numbers from FT (September 2006)
  - 80% to 90% of P2P traffic
    - Only 10% to 20% of P2P traffic due to BitTorrent
    - The rest is due to Emule

- 78% of P2P traffic in Japan in 2008
More Recent Results

- Sandvine (Fall 2010)
  - P2P is dominating *upstream* traffic
  - Real-time entertainment (*audio and video streaming*) is dominating the *downstream* traffic
    - But, P2P share has significantly increased since 2009 (doubled in some regions)
Lessons Learned From the Past

- **Specific events** might significantly impact popularity of P2P protocols
  - Disconnection of popular services
    - Suprnova, mininova, ThePirateBay
  - Specific laws

- This has always been a **transient** impact
Why to Study P2P (Old Version)

P2P represents most of the Internet traffic

Yes, but it is for illegal contents. P2P applications are evil

Don’t you think there is a need for such a service

Yes, but people should pay for the service and we need to keep control on it

And, in this case which techno will you use to reach millions of users without huge distribution costs

P2P
Why to Study P2P (New Version)

- BitTorrent is super fast to distribute contents
  - Start to be used by several big companies

- Twitter is using Murder to update Twitter servers (July 2010)
  - 75x faster
Murder

Without Murder  With Murder

Credit: Larry Gadea
Murder Performance

Credit: Larry Gadea
Tutorial Outline

- File distribution
  - BitTorrent
  - Min Dist Time
- Distributed database
  - DHTs
  - Chord
- File sharing
  - Napster
  - Gnutella
  - FastTrack
  - Overnet
- P2P file sharing security
  - Attacks on P2P systems
  - Attacks from P2P systems
- VoIP
- Anonymity
- Live video streaming
  - Measurement results
  - Theory
- On-demand streaming
BitTorrent

- A Peer-to-Peer Content Distribution Protocol/Program

- Developed by Bram Cohen in 2001
  - Bram grew up in upper west side of Manhattan, NYC

- First version written in Python
BitTorrent Overview

1. Get a .torrent file that contains the address of the tracker.
2. Get a random peer set.
3. Start downloading.
4. Exchange pieces with other peers.
5. Complete the download.

Initial Seed

Web server

Tracker

coolContent.xvid

P1 P2 P3
BitTorrent Specificities

- Unlike any other P2P protocol, there is one session per content
  - A session is called a torrent
  - Torrents are content based

- Torrents are independent
  - You get no benefit from previous or current torrents → Private torrent
  - No enforcement to stay as a seed
Pieces and Blocks

- Content is split into pieces, which are split into blocks
Pieces and Blocks

- **Pieces**
  - The *smaller unit of retransmission*
  - Typically 256/512/1024/2048 kByte
  - Size adapted to have a reasonably small .torrent file

- **Blocks**
  - 16kB (hard coded)
  - Used for pipelining
    - Always 5 requests pending
.torrent file

- .torrent file encoded using bencoding
  - Info key
    - Length on the content in bytes
    - md5 hash of the content (optional)
      - Not used by the protocol
      - pieces SHA-1 hash are enough
    - File Name
    - Piece length (256kB, 512kB, 1024kB, etc.)
    - Concatenation of all pieces SHA-1 hash
.torrent file

- .torrent file encoded using bencoding
  - Announce URL of the tracker (HTTP)
    - Possibility of announce list for backup trackers
      - See http://www.bittorrent.org/beps/bep_0012.html
  - Some optional fields
    - Creation date, comment, created by
Piece Selection

- 4 policies
  - Strict priority
  - Random first piece
  - Local rarest first
  - Endgame mode
**Strict Priority**

- Once a block of a piece has been requested, request **all the other blocks of the same piece** before a block of any other piece.

**Rationale**

- **Pieces are the unit of replication**
  - It is important to download a piece as fast as possible, only complete pieces can be retransmitted.
Random First Piece

- For the first 4 downloaded pieces, the pieces are selected at random.

Rationale
- Rare pieces may be slower to download.
- A peer without a piece cannot reciprocate.
Local Rarest First

- Download first the pieces that are **rarest** in the peer set of the peer

- **Rationale**
  - Cannot maintain the state for all peers
Endgame Mode

- When all blocks are either received or have pending requests, request all not yet received blocks to all peers. Cancel request for blocks received.

- Rationale
  - Prevent the termination idle time
Sending Chunks: tit-for-tat

- Alice sends chunks to **four neighbors** currently sending her chunks *at the highest rate*
  - re-evaluate top 4 every 10 secs

- Every 30 seconds: randomly select another peer, starts sending chunks
  - newly chosen peer may join top 4
  - "optimistically unchoke"
BitTorrent: Tit-for-tat

(1) Alice “optimistically unchokes” Bob
(2) Alice becomes one of Bob’s top-four providers; Bob reciprocates
(3) Bob becomes one of Alice’s top-four providers

With higher upload rate, can find better trading partners & get file faster!
Minimum distribution time?

File size = $F$

Server

$u_s$  $u_1$  $u_2$

$d_1$  $d_2$

$u_L$  $d_L$

Abundant Bandwidth
Let’s First Get a Lower Bound

How fast can the server pump out the file?

$$T_{\text{min}} \geq F/u_s$$
Let’s First Get a Lower Bound

How fast can client $i$ receive the file?

$$T_{\text{min}} \geq \frac{F}{d_i}$$

Thus

$$T_{\text{min}} \geq \frac{F}{d_{\text{min}}}$$
Let’s First Get a Lower Bound

How fast can the servers and peers upload the file to L peers?

\[ T_{\text{min}} \geq \frac{LF}{u_s + u_1 + \cdots + u_L} \]

Define \( u_T = u_s + u_1 + \cdots + u_L \)
Putting it Together

\[ T_{\text{min}} \geq \max\{F/u_s, F/d_{\text{min}}, LF/u_T\} \]

But is there a scheduling policy that achieves the lower bound?

Yes when bits are fluids!
In Summary

\[ T_{\text{min}} = \max\{F/u_s, F/d_{\text{min}}, LF/(u_s+u_1+...+u_L)\} \]
P2P vs. Client-Server

Minimum Distribution Time

Peer-Assisted

Client-Server

L
BitTorrent Use Case

- BitTorrent is for efficient file replication

- It is not for
  - Content localization
    - May be hard to find a content
  - Content availability
    - Torrents can die fast, no incentive to keep a torrent up
  - Both issues are important, but orthogonal
Tutorial Outline

- File distribution
  - BitTorrent
  - Min Dist Time

- Distributed database
  - DHTs
  - Chord and Pastry

- File sharing
  - Napster
  - Gnutella
  - FastTrack
  - Overnet

- P2P file sharing security
  - Attacks on P2P systems
  - Attacks from P2P systems

- VoIP

- Anonymity

- Live video streaming
  - Measurement results
  - Theory

- On-demand streaming
Distributed Hash Table (DHTs)

- Unstructured P2P systems can take a long time to locate an object.

**Motivation behind DHTs**
- Find objects faster
- Find rare content

**Idea:**
- Assign particular nodes to hold particular objects (or pointers to it)
- When a node wants that object, go to the node that is supposed to have it.
DHT object assignment

- object id’s and node id’s
  - e.g., id’s are 128 bit strings
- example:
  - Node randomly chooses id
  - Object id is a 128-bit hash of object
- Assignment scheme
  - Assign object to node that has the closest id to its own
Easy case: each node knows about all other nodes

- Suppose each peer has local table that includes all peers:
  - Node ID, IP address

- Want some object

- Solution:
  - Locally calculate $h = h(\text{object})$
  - Find node ID in table that is closest to $h$
  - Obtain corresponding IP address
  - Ask that node for the object

- Solution is very fast: $O(1)$ messages

- But
  - P2P systems have millions of nodes: $O(N)$ neighbors, which come and go
DHT overlay

- Organize nodes in an overlay
- Every node knows about some other nodes
  - Neighbors
- Route queries for objects over overlay
- When node receives query for object with ID h:
  - Forwards the query to neighbor whose ID is closest to h
- Can we design DHT so:
  - Nodes don’t have too many neighbors?
  - Eventually find node responsible for h?
  - Not too many hops?
Circle DHT

- Overlay network is a circle
- Node's successor in circle is node with next largest id
  - Each node has two neighbors: successor and predecessor
  - Knows IP address of its neighbors
- Object is stored in closest successor
Circular DHT (2)

O(N) messages on avg to resolve query

Who's resp for file 1110

I am

file 1110 stored here

Note: no locality among neighbors
Circular DHT (3)

- Need also to replicate object to handle node joins and leaves

- Store object on two closest successors

- If either leaves, copy object to next successor
Circular DHT (4)

- Overlay is actually a circle with small chords for tracking s successors

- average # of messages to find key is $O(N)$

Can we do better?
Chord

- Frequently cited DHT
- Basically circle with chords
- Each node has \( \log(N) \) neighbors (chords):
  - \( i^{th} \) neighbor of node \( s \) has the ID that is equal to \( s+2^i \) or is the next largest ID
  - Chord from \( s \) to \( s+1, s+2, s+4, s+8, s+16, \ldots \)
- Routing: send query with hash \( h \) to largest neighbor \( \leq h \)
- Node’s set of a neighbors is an overlay forwarding table.
  - Called fingering table in Chord jargon
Chord example

ID space = [0,99]
Tutorial Outline

- File distribution
  - BitTorrent
  - Min Dist Time
- Distributed database
  - DHTs
  - Chord and Pastry
- File sharing
  - Napster
  - Gnutella
  - FastTrack
  - Overnet
- P2P file sharing security
  - Attacks on P2P systems
  - Attacks from P2P systems
- VoIP
- Anonymity
- Live video streaming
  - Measurement results
  - Theory
- On-demand streaming
Technical issues for File Sharing

- **Search**
  - Centralized or decentralized
  - Decentralized: unstructured or DHT

- **Download**
  - Multiple sources (swarming)
  - Peers behind NATs

- **Security**
  - Susceptibility to attacks

- **Incentives**
  - Free-riding
Killer file sharing deployments

- **Napster**
  - disruptive; proof of concept
- **Gnutella**
  - open source
- **KaZaA/FastTrack**
  - Today more KaZaA traffic then Web traffic!
- **eDonkey / Overnet**
  - Uses a DHT
- **BitTorrent + search (eg PirateBay)**
  - Content distribution

Is success due to massive number of servers, or simply because content is free?
Napster

- First successful P2P file-sharing app
- a “disruptive” application/technology?

**history:**
- **5/99:** Shawn Fanning (freshman, Northeasten U.) founds Napster Online music service
- **12/99:** first lawsuit
- **3/00:** 25% UWisc traffic Napster
- **2/01:** US Circuit Court of Appeals: Napster knew users violating copyright laws
- **7/01:** # simultaneous online users:
  - Napster 160K, Gnutella: 40K, Morpheus (KaZaA): 300K
original “Napster” design
1) when peer connects, it informs central server:
   - IP address
   - content
2) Alice queries for “Hey Jude”
3) Alice requests file from Bob
P2P: problems with centralized directory

- Single point of failure
- Performance bottleneck
- Copyright infringement

file transfer is decentralized, but locating content is highly centralized
Query flooding

- fully distributed
  - no central server

- used by Gnutella

- Each peer indexes the files it makes available for sharing (and no other files)
**Query flooding**

- Query message sent over existing TCP connections
- Peers forward Query message
- QueryHit sent over reverse path

**Scalability:** limited scope flooding

**File transfer:** HTTP
Distributed Search/Flooding
Distributed Search/Flooding
Exploiting heterogeneity: FastTrack/Kazaa

- Each peer is either a supernode or assigned to a supernode.
  - TCP connection between peer and its supernode.
  - TCP connections between some pairs of supernodes.

- Supernode tracks the content in all its children.
FastTrack Operation

ON = ordinary node
SN = super node

Each SN maintains a local index
Tutorial Outline

- File distribution
  - BitTorrent
  - Min Dist Time

- Distributed database
  - DHTs
  - Chord and Pastry

- File sharing
  - Napster
  - Gnutella
  - FastTrack
  - Overnet

- P2P file sharing security
  - Attacks on P2P systems
  - Attacks from P2P systems

- VoIP

- Anonymity

- Live video streaming
  - Measurement results
  - Theory

- On-demand streaming
Attacks in P2P file systems

- **Attacks on P2P systems:**
  ![Diagram of attacks on P2P systems]

- **Attacks from P2P Systems:**
  ![Diagram of attacks from P2P systems]
How can a P2P app be attacked?

- Attack content
- Attack overlay routing
- Attack distributed index
- Exploit software vulnerability
- Flood peers
- Anything else?
Attacks on P2P: Today

Two types:
- File corruption: pollution
- Index poisoning

Investigated two networks:
- FastTrack/Kazaa
  - Unstructured P2P network
- Overnet
  - Structured (DHT) P2P network
  - Part of eDonkey
File Pollution

original content

polluted content

pollution company
File Pollution

pollution company

file sharing network

pollution server

pollution server

pollution server
File Pollution

Unsuspecting users spread pollution!
File Pollution

Unsuspecting users spread pollution!
Index Poisoning

- **Index**
- **Title**: bigparty, smallfun, heyhey
- **Location**: 123.12.7.98, 23.123.78.6, 234.8.89.20
Index Poisoning

<table>
<thead>
<tr>
<th>title</th>
<th>location</th>
</tr>
</thead>
<tbody>
<tr>
<td>bigparty</td>
<td>123.12.7.98</td>
</tr>
<tr>
<td>smallfun</td>
<td>23.123.78.6</td>
</tr>
<tr>
<td>heyhey</td>
<td>234.8.89.20</td>
</tr>
<tr>
<td>bighit</td>
<td>111.22.22.22</td>
</tr>
</tbody>
</table>

Network diagram showing index poisoning with IP addresses and titles.
overpeer provides powerful solutions for content owners and creators to protect digital media assets

Overpeer, a Loudeye Company, is the leading worldwide provider of digital media data mining, anti-piracy and promotional solutions. Overpeer provides creators and copyright owners with valuable market intelligence on the unauthorized digital distribution of their content assets as well as anti-piracy and promotional tools to protect content and capitalize on previously untapped revenue opportunities across file sharing networks. As a result, our partners gain valuable insight into the unauthorized digital distribution of their content assets and can access to important tools to make strategic business decisions around digital distribution.

Overpeer provides:
- Powerful data mining and analytical tools and comprehensive information on
Index Poisoning in FastTrack and Overnet

- **FastTrack/Kazaa**
  - Non-DHT with supernodes
  - Advertise to supernodes
    - \((target\_song, \text{bogus\_IP})\)
      - for many bogus IP’s, many versions of target\_song

- **Overnet/E-donkey**
  - Advertise
    - \((\text{hash\_target\_keyword}, \text{bogus\_version\_id})\)
The Inverse Attack

- Attacks on P2P systems:

- But can also exploit P2P systems for DDoS attacks against innocent host:
DDoS Attacks From: Poison Index

Example

- target_IP = www.poly.edu
  - Advertise records:
    (Madonna_hit, target_IP:target_port)

Users attempt to download popular title

  - Generates fully-open TCP connections from user peers to target
  - Nastier than syn-flood DDoS attack
DDoS Poison Index: Connections

![Graph showing the number of TCP connections over time, with two lines: one for 'Total' and another for 'ESTABLISHED'.]
DDoS Poison Index: Connection Durations
DDoS Attacks in DHT: Poison Routing Tables

- Example
  - Routing advertisements: \((\text{node\_ID, target\_IP})\)
  - Many nodes will absorb advertisement
- Query, publish, overlay messages arrive at poisoned node
  - Some are directed at target
DDoS Poison Routing: UDP Bandwidth
DDoS Poison Routing: IP sources
Summary

- Pollution and Poison attacks
- Can attack arbitrary title
  - Developed methodology to evaluate success of attack without downloading content
  - Leads to distributed blacklisting scheme
- Can also DDoS attack arbitrary node with poisoning
  - Index poisoning
  - Routing poisoning
Blacklisting

- Assign **reputations** to /n subnets
  - Bad reputation to subnets with large number of advertised copies of any title
- Obtain reputations locally;
- share with distributed algorithm
- **Locally blacklist** /n subnets with bad reputations
Blacklisting: More
Tutorial Outline

- File distribution
  - BitTorrent
  - Min Dist Time

- Distributed database
  - DHTs
  - Chord and Pastry

- File sharing
  - Napster
  - Gnutella
  - FastTrack

- P2P file sharing security
  - Attacks on P2P systems
  - Attacks from P2P systems

- VoIP

- Anonymity

- Live video streaming
  - Measurement results
  - Theory

- On-demand streaming
P2P Voice-over-IP?

Skype - Free Internet telephony that just works - Microsoft Internet Explorer

Download Skype now. It's free.

Download the latest Skype for Windows if you are using an earlier version than 1.0.0.100.

Skype is free Internet telephony that just works. Skype is for calling other people on their computers or phones. Download Skype and start calling for free all over the world.

Call ordinary phones from Skype with SkypeOut. Top up on SkypeOut credit and use the Skype programme to call ordinary phones all over the world. Not free, but pretty cheap.
Skype Today

- Acquired by eBay in Sept 2005 for more than $2 billion
- Today about 4 million concurrent users
- Video in beta test.
- Why so popular?
  - Timing
  - Free
  - Often outstanding voice quality
  - Simple interface
  - NAT solution
Relays are peers in Skype!
Tutorial Outline

- File distribution
  - BitTorrent
  - Min Dist Time
- Distributed database
  - DHTs
  - Chord and Pastry
- File sharing
  - Napster
  - Gnutella
  - FastTrack
  - Overnet
- P2P file sharing security
  - Attacks on P2P systems
  - Attacks from P2P systems
- VoIP
- Anonymity
- Live video streaming
  - Measurement results
  - Theory
- On-demand streaming
Anonymity (Privacy)

- Suppose you are surfing the Web.
- You don’t want Web site to know your IP address.
- And you don’t want your ISP to sniff your traffic.
- Possible solution: use anonymizing proxy.
What is TOR?

Tor – The Onion Router

- A **free software** implementation of second-generation onion routing.
- Protect **privacy and security** on the Internet by keeping anonymity of user.
- Allows users to bypass Internet censorship through its **volunteer-based** distributed infrastructure.
Background

- Originally sponsored by the US Naval Research Laboratory
- From 2004 to 2006 was supported by EFF (Electronic Frontier Foundation)
- Tor software is now developed by the Tor Project

Tor VS. One-hop proxy

- Commercial one-hop proxy: Hotspot Shield, Anonymizer, Anonymouse
Tor VS. One-hop proxies

- Tor

```
Client

Server

Client
```

```
Tor Relay

Tor Relay

Tor Relay

Server

Tor Relay

Tor Relay

Tor Relay
```

```
Tor Relay

Tor Relay

Tor Relay

Server

Tor Relay

Tor Relay

Tor Relay
```

```
Tor Relay

Tor Relay

Tor Relay

Server

Tor Relay

Tor Relay

Tor Relay
```

```
Tor Relay

Tor Relay

Tor Relay

Server

Tor Relay

Tor Relay

Tor Relay
```
Tor on Client Side

App -> HTTP -> Privoxy/polipo -> SOCKS -> Tor

App -> SOCKS -> Tor
Tor on Client Side - Tor Bundle

A typical tor installation Bundle includes:
- **Tor**: command line interface.
- **Vidalia**: GUI for tor. (built using Qt framework)
- **Polipo**: a web proxy.
- **Torbutton**: Firefox add-on.
Tor on Client Side - Vidalia
Tor on Client Side - Vidalia

Supported apps:

- Websites
- IM
- SSL
- IRC
- POP, IMAP
- Other (TCP)
How Tor Works?

- Get address list from centralized directory server.
- Tor nodes are connected via TLS(ssl).

Image courtesy https://www.torproject.org
How Tor Works?

- **Clients** build circuits through a network of decrypting relays.
- One circuit can be used to relay multiple TCP streams.

Image courtesy [https://www.torproject.org](https://www.torproject.org)
How Tor Works?

- Entry node changed every ten minutes if needed
- Alice can run as a tor node to benefit others.

Image courtesy https://www.torproject.org
Anonymity: Summary

- Proxy server + encryption can provide a degree of anonymity
  - But what if you can’t trust the proxy server?
- Chaining proxy servers provide more protection.
  - As long as the server don’t collude
Tutorial Outline

- File distribution
  - BitTorrent
  - Min Dist Time
- Distributed database
  - DHTs
  - Chord and Pastry
- File sharing
  - Napster
  - Gnutella
  - FastTrack
  - Overnet
- P2P file sharing security
  - Attacks on P2P systems
  - Attacks from P2P systems
- VoIP
- Anonymity
- Live video streaming
  - Measurement results
  - Theory
- On-demand streaming
On-Demand Internet Video

- Youtube
  - 1.167 billion videos, streamed in January 07 by 55 million people
  - User generated content

- Viacom (MTV)
  - 264 million videos, streamed by 19 million
  - “produced” content

- Joost
  - Founded by Kazaa/Skype people
  - P2P VoD
Live Internet Video: Vision

- Thousands of live channels
  - TV from hundreds of countries
  - Independent artists
  - Lectures and events
  - User generated from Webcams
  - Hand-held wireless devices

- High-quality image and sound
  - 1Mbps+

- Channel viewed by thousands to millions of users
Live Video Challenge

- How does a popular site distribute
  - a million streams
  - at 1 Mbps
  - in cost effective manner

- From server to 1M clients: 1 terabit/s
  - Cost prohibitive

- Naturally, consider P2P
P2P IPTV: success stories

- **Coolstream:**
  - 4,000 simultaneous users in 2003

- **PPLive:**
  - 200,000 users at 400-800 kbps for 2006 Chinese New Year
  - Aggregate rate of 100 Gbps

- **ppstream, UUsee, and many others**
Basic idea P2P live streaming

tracker

obtain list of peers

peer

trade chunks

Source of video

peer

trade chunks

peer

peer

peer

peer

peer

peer
Measurement Methodologies

- Protocol analysis
  - PPLive, ppstream

- Active crawling
  - Visit all peers in a channel every minute
  - Learn IP addresses of active peers
  - Obtain buffer maps from all active peers

- Passive sniffing
  - Monitors at residences and university campuses
  - Sniff and analyze traffic
Measurement Aparatus

- 30 peers per Linux box
- 5-10 boxes
From Crawler: 
Popular Channel CCTV3

![Graph showing the number of active peers per day of the week over a period of several months. The graph has a y-axis labeled "# of active peers" with values ranging from 0 to 2500, and an x-axis labeled "Data(noon)" with days of the week labeled from Saturday to Sunday. The data shows fluctuations in the number of active peers throughout the week and month.]
PPLive: Chinese New Year 2006

![Graph showing the number of peers over time on Jan 28 and Jan 29. The graph has a logarithmic scale on the y-axis. The x-axis represents time in hours from 4 to 24. The peak number of peers is observed on Jan 28.]
Upload and download rates: Home, Popular
Upload and download rates: Campus, Popular Channel

![Graph showing upload and download rates over time](image_url)
References

- B. Cohen, Incentives build robustness in BitTorrent, *Proc. of IPTPS 2003*
- A. Rowstron and P. Druschel, Pastry: Scalable, Decentralized, Object Location and Routing for Large-scale Peer-to-peer Systems, *Proc. of IFIP/ACM Int. Conf. on Distributed Systems Platforms (Middleware)'02*
- P. Maymounkov and D. Mazières, Kademlia: A Peer-to-peer Information System Based on the XOR Metric, *IPTPS 2002*
- J. Liang, R. Kumar, Yongjian Xi, K.W. Ross, Pollution in P2P File Sharing Systems, *Infocom 05*
- C. Huang, J. Li, K.W. Ross, Peer-Assisted VoD: Making Internet Video Distribution Cheap, *IPTPS'07*
Thank you!