



# *Peer-to-peer filetransfer protocols and IPv6*

János Mohácsi  
NIIF/HUNGARNET  
TF-NGN meeting, 1/Oct/2004

# Motivation

- ◆ IPv6 traffic is  $< \sim 1/100$  of IPv4 traffic
  - ◆ (dominant protocols on IPv6: IRC, HTTP, bursty videostreaming)
- ◆ Where is missing gap in IPv6 deployment?
  - ◆ End users?
- ◆ Goal:
  - ◆ Attract users to use IPv6:
    - ◆ Increase the IPv6 enabled HTTP server – controlled IPv6 HTTP proxies
    - ◆ Distribute USENET News via IPv6
    - ◆ P2P applications – with IPv6 you can have e2e connectivity
      - ◆ p2p messaging – jabber
      - ◆ p2p filesharing?

# *P2P filetransfer protocols*

- ◆ Zillions of p2p filesharing programs/protocols – popular ones:
  - ◆ Napster – hardcoded IP – 4 bytes ☹
  - ◆ Gnutella – hardcoded IP – 4 bytes ☹
  - ◆ DCC – IP is unsigned long in a document ☹, other documents says ascii string?
  - ◆ Edonkey – hardcoded IP – 4 bytes ☹
  - ◆ Bittorrent – IP/hostname is bencoded string ☺ – candidate for IPv6 filesharing
  - ◆ Gnunet – designed to be IPv6 (any transport - even SMTP) compatible ☺
  - ◆ ....



## *Why Bittorrent is developed?*

- ◆ How do we transfer a piece of data quickly to a group of people interested in it.
  - ◆ Web server
  - ◆ Distributors
  - ◆ Xzy distribution patches
- ◆ One possible solution to problem above: Bit-Torrent based file swarming
  - ◆ One of the first legitimate p2p filesharing network (s)

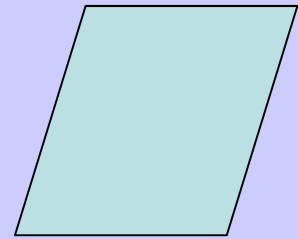


# Overall Architecture

Web  
Server

Tracker

Web page  
with link  
to .torrent



.torrent

A

C

Peer

[Leech]

Download

B

Peer

[Leech  
]

Peer

[Seed]

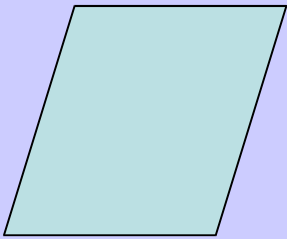


# Overall Architecture

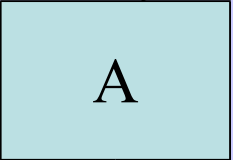
Web Server

Tracker

Web page with link to .torrent



Get-announce



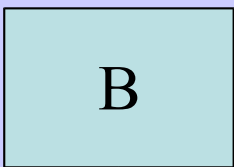
Peer

Peer

[Leech]

[Seed]

Download



Peer

[Leech

]

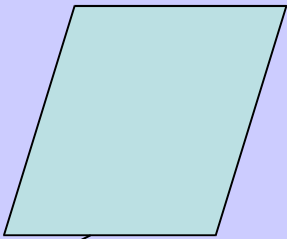


# Overall Architecture

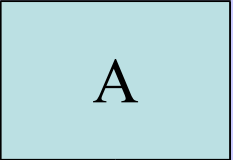
Web Server

Tracker

Web page with link to .torrent



Response-peer list



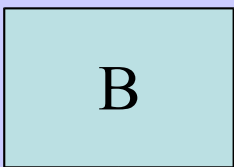
Peer

Peer

[Leech]

[Seed]

Download



Peer

[Leech

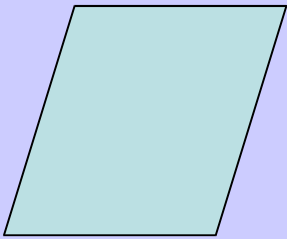
]



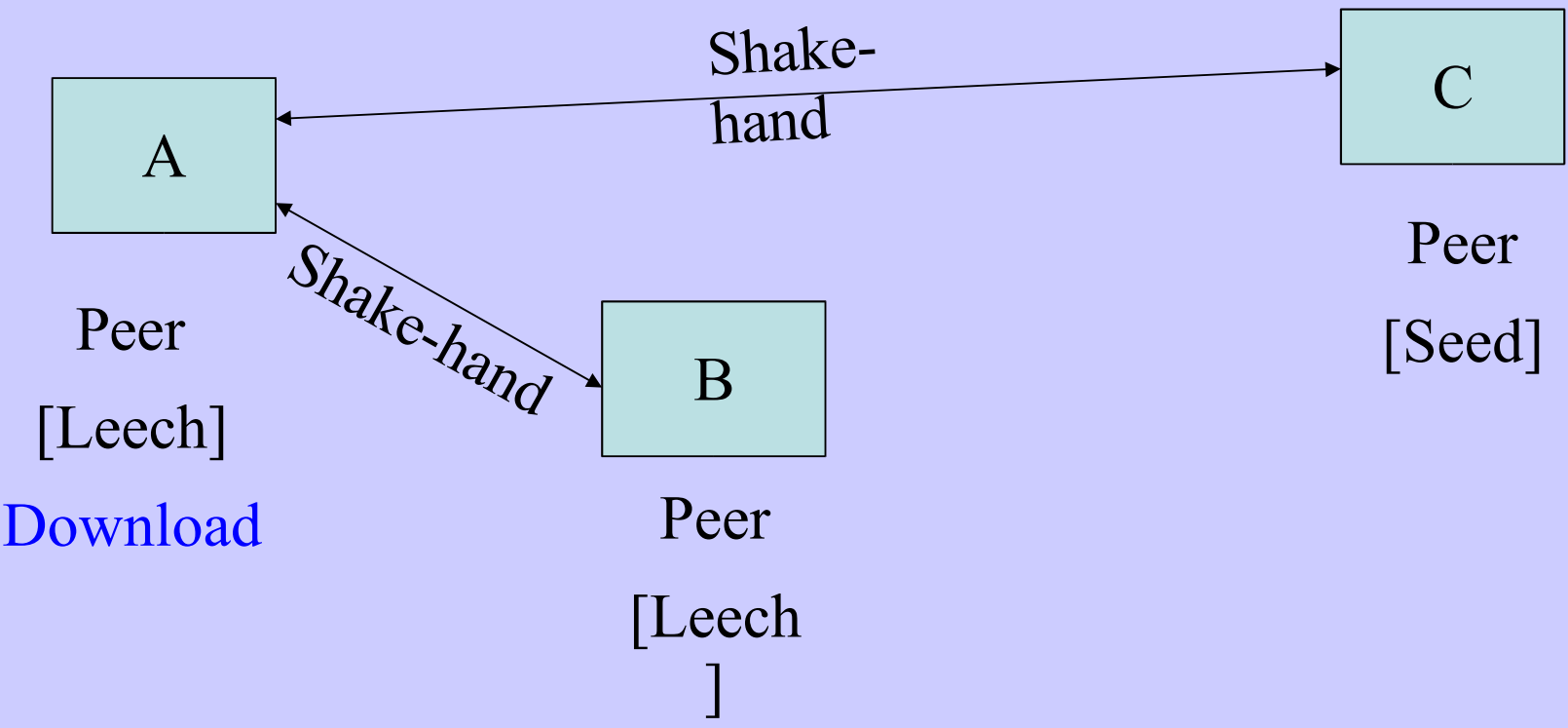
# Overall Architecture

Web page with link to .torrent

Web Server



Tracker

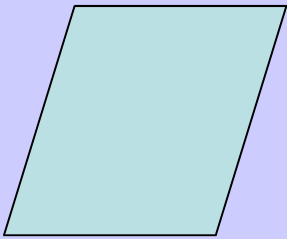




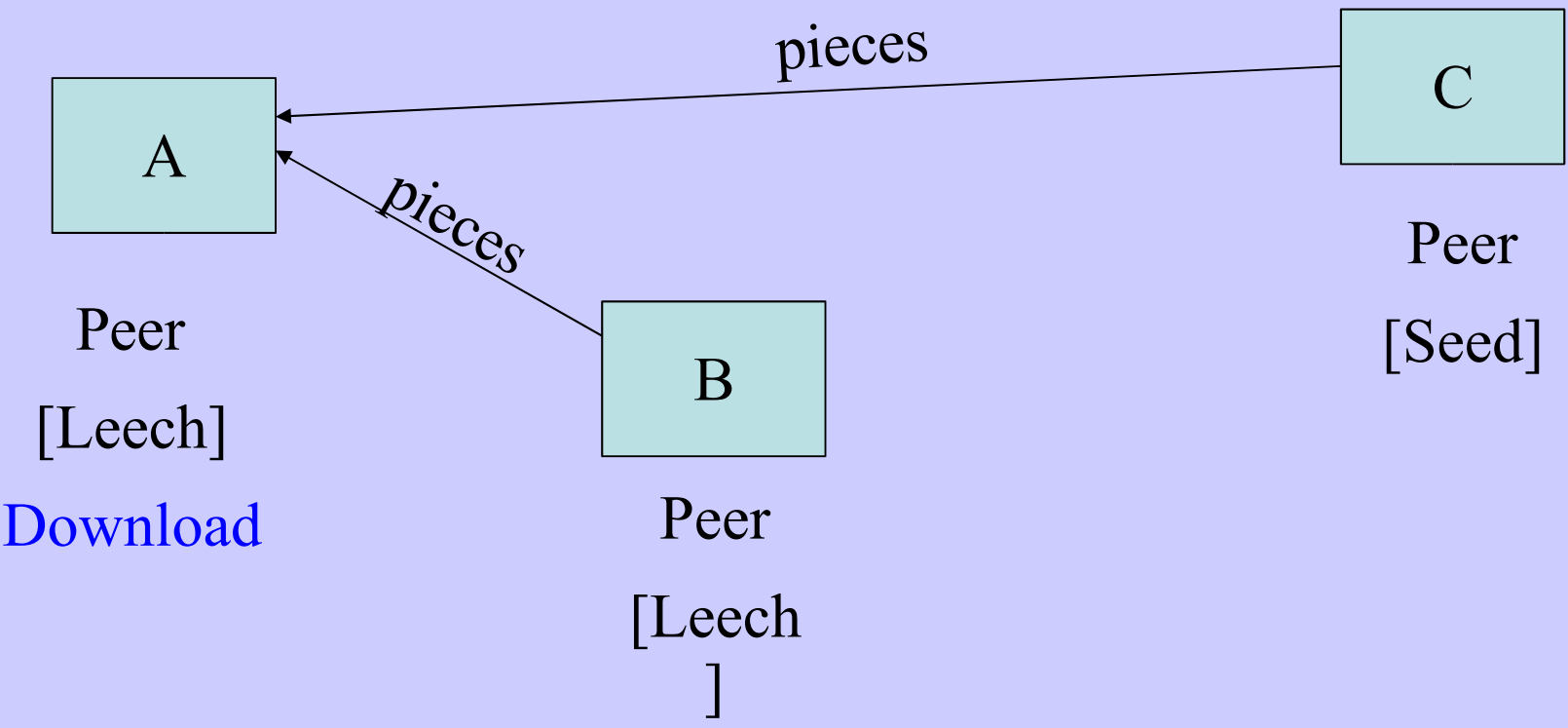
# Overall Architecture

Web page with link to .torrent

Web Server



Tracker

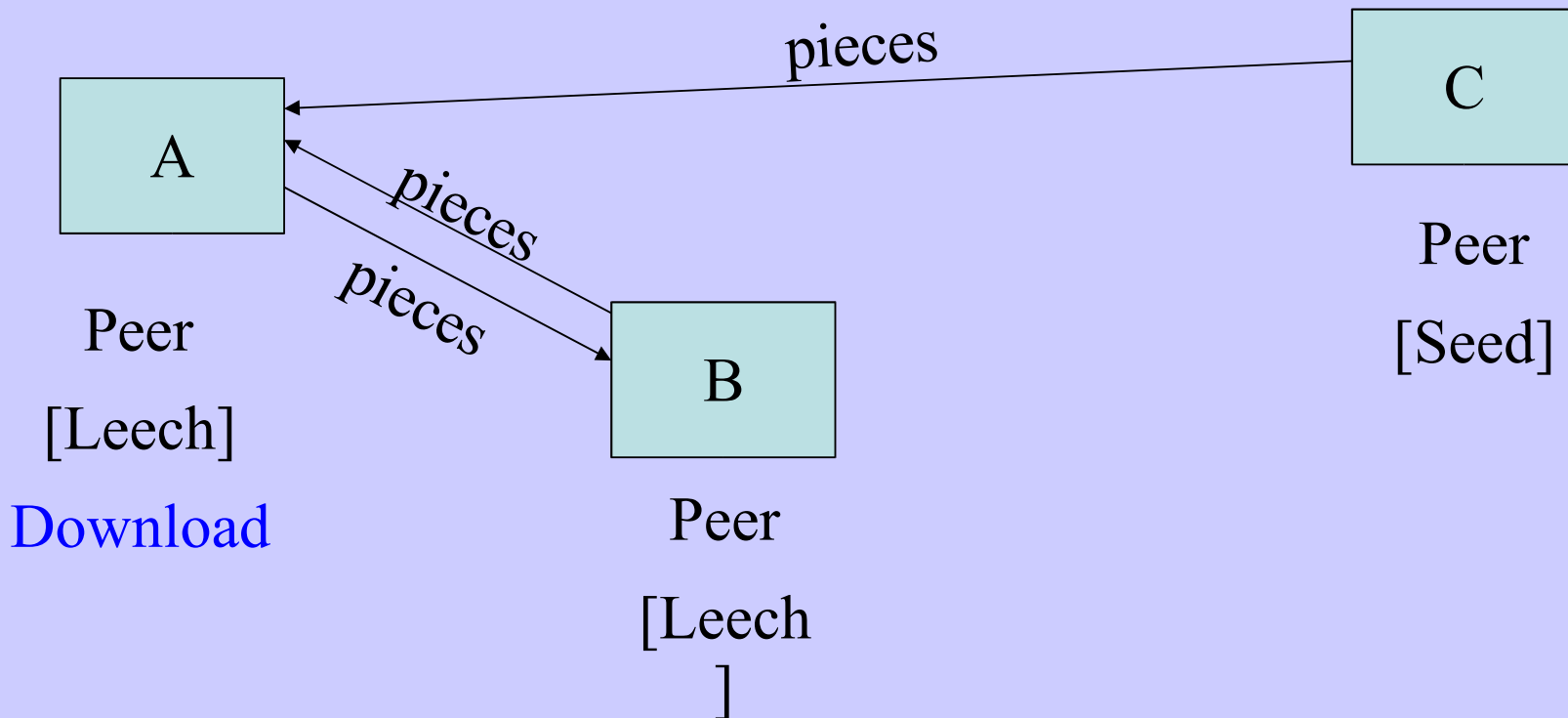
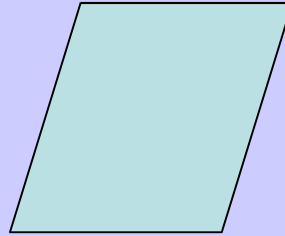
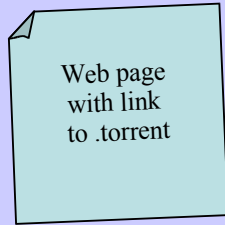




# Overall Architecture

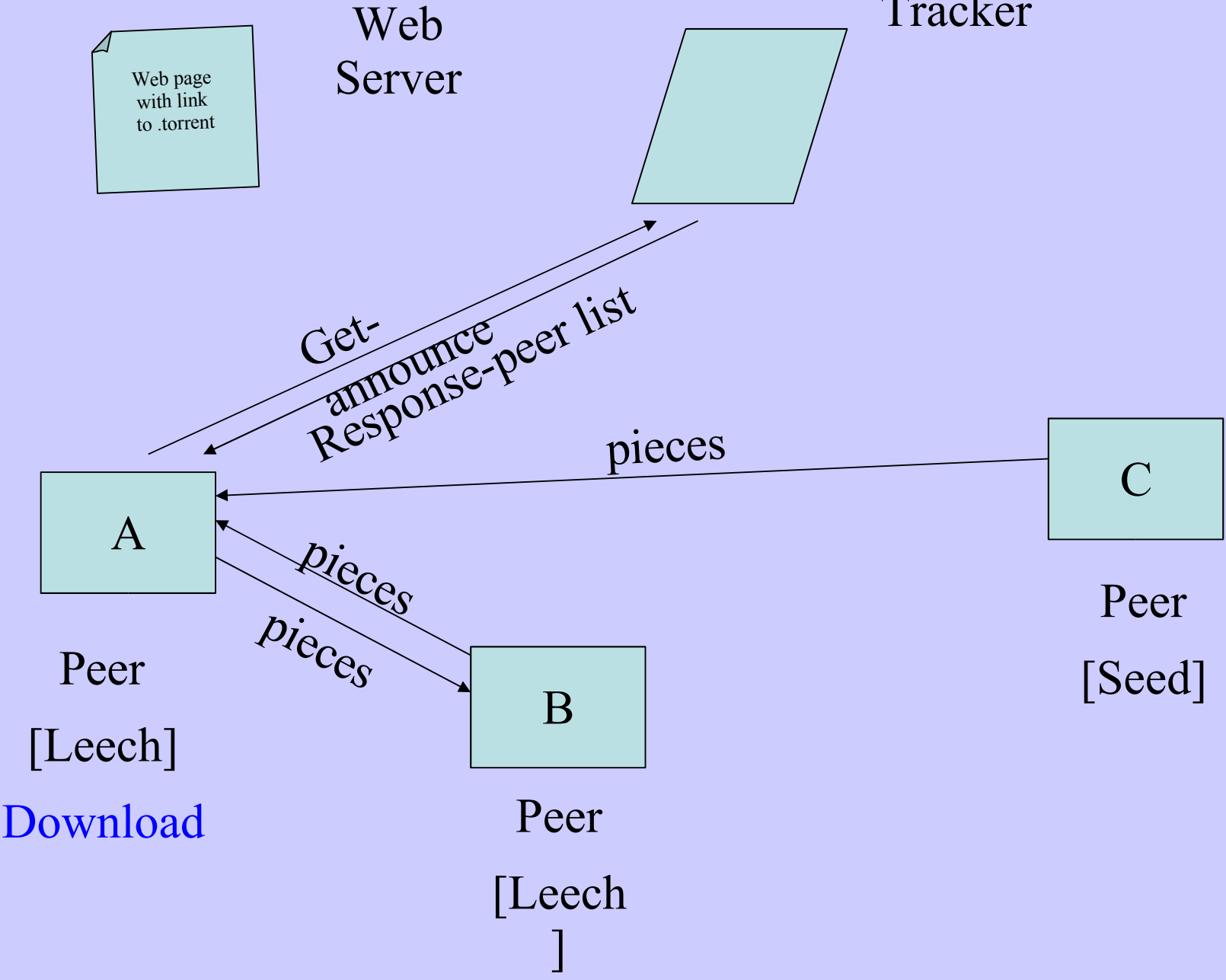
Web  
Server

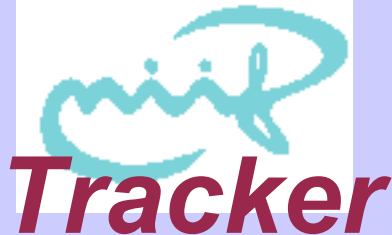
Tracker





# Overall Architecture





- ◆ Peer cache
  - ◆ IP/hostname, port, peer id - bencoded
- ◆ State information
  - ◆ Completed
  - ◆ Downloading
- ◆ Returns random list



# *Peer Operation*

- ◆ File space allocated
- ◆ Connect to peers
- ◆ Bitfield
  - ◆ have(1) and not have(0)
- ◆ Have <piece>
  - ◆ Advertise pieces
- ◆ Interest /Not\_interested
  - ◆ Expressing interest in the pieces published by the peer
- ◆ Requesting for a chunk
  - ◆ Request <index, offset, length>

# *Peer Operation*

- ◆ Choking algorithm
  - ◆ Choke/Unchoke
  - ◆ Preferred peers
  - ◆ Optimistic unchoke
  - ◆ Snubbing behavior
    - ◆ Prevented by Anti-snubbing.
- ◆ Upload to interested peers who are not choking.



## *Peer Operation*

- ◆ Verify on receiving complete piece
- ◆ Endgame Behavior
  - ◆ Cancel



# Client

BitTorrent T-0.3.7 (BitTornado) download

Details Advanced Prefs About

Time elapsed / estimated : 0 min 00 sec / Failed!

Download to:

Download rate:	Downloaded: 0.00 MiB	Share rating:
Upload rate: 0 kB/s	Uploaded: 0.00 MiB	

Pause Close

Settings for: automatic

Upload rate (kB/s) 0 0 500

Max uploads 4 4 100

0 kB/s means unlimited. Tip: your download rate is proportional to your upload rate

# Strengths

- ◆ Better bandwidth utilization
  - ◆ Very high speeds in spite written in python
    - ◆ Up to 7 MB/s – possible (in LAN)
    - ◆ In real test bittorrent download: ~ 700 KB/s, FTP download : ~ 1300 KB/s
- ◆ Limit free riding – tit-for-tat
- ◆ Limit leech attack – coupling upload & download
- ◆ Spurious files not propagated
- ◆ Ability to resume a download



# *Drawbacks*

- ◆ Small files – latency, overhead
- ◆ Random list of peers - naive
- ◆ Scalability
  - ◆ Millions of peers – Tracker behavior (uses 1/1000 of band width)
  - ◆ Single point of failure – possible resolution: multitracker
- ◆ Robustness
  - ◆ System progress dependent on nature of seeds (and peers)
  - ◆ Malicious attacks and leeches.
- ◆ No real search



## *Bittorrent adventure with IPv6*

- ◆ Original Bittorrent client version 3.3 – patch to support IPv6 – available at:
  - ◆ <http://6net.niif.hu/index.php?mn=3&sm=9&lg=en>
  - ◆ Correct patch supports wildcard IPv4/IPv6 binding
- ◆ Bittorrent 3.4 introduces changes in the tracker protocol:
  - ◆ Key: to easier find the file/peer on multifile trackers
  - ◆ Compact=1: ask tracker to reply with compact not bencoded IP+port (6 byte binary) – no longer IP/transport neutral.

## *Bittorrent adventure with IPv6/2*

- ◆ Shadow bittorrent client supported IPv6
- ◆ Bittornado (successor of shadow bittorrent client) implements Bittorrent 3.4 changes
  - ◆ no longer IP neutral.
  - ◆ There is a patch available at:  
<http://6net.niif.hu/index.php?mn=3&sm=9&lg=en>  
to disable compact=1 requests at peers
- ◆ ABC bittorrent client (Windows) supported IPv6 on Linux – based on shadow bittorrent client
  - ◆ Does not support IPv6 on Windows – no IPv6 support in compiled Windows Python versions
    - ◆ cygwin python might work – untested yet
    - ◆ Binary python 2.4 for Windows will support IPv6

# *Bittorrent adventure with IPv6/3*

- ◆ Problem with the current implementation
  - ◆ Tracker select randomly the peers:
    - ◆ IPv4 only peer cannot connect to IPv6 peer
    - ◆ Solution1:
      - ◆ changes in tracker: give IPv6 peers to IPv6 peers
    - ◆ Solution2:
      - ◆ changes in tracker: store names in peer cache no IP addressess.
  - ◆ The author has different view of developing the protocol
    - ◆ No real plan for IPv6 – adding another complexity
    - ◆ No acceptance for keeping names in tracker peer cache – developer claims DNS is unreliable
    - ◆ The development of original Bittorrent client seems to be stalled

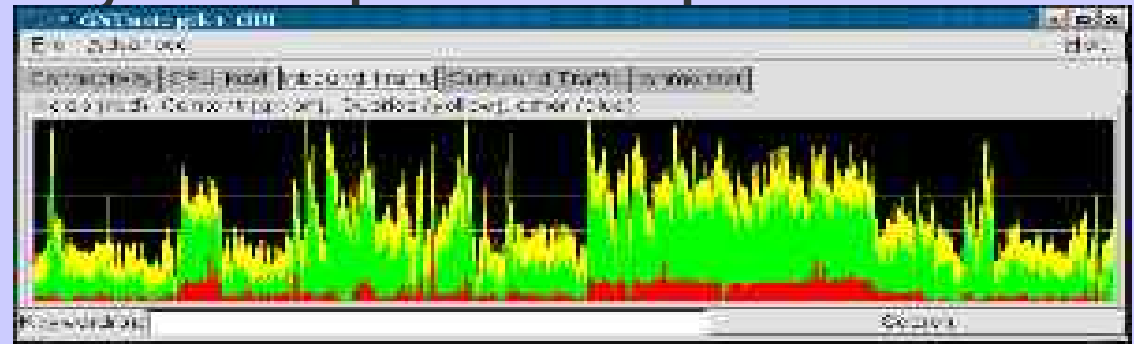


## *Interesting links*

- ◆ Official site: <http://bitconjurer.org/BitTorrent>
- ◆ Good client:
  - ◆ <http://www.bittornado.com>
  - ◆ Patches:  
<http://6net.niif.hu/index.php?mn=3&sm=9&lg=en>
- ◆ To experiment with torrents via IPv6:
  - ◆ <http://6net.niif.hu/index.php?mn=3&sm=10&lg=en>
- ◆ BitTorrent FAQ: <http://btfaq.com>

# GNUNET

- ◆ Designed to be any transport compatible
  - ◆ TCP, UDP
  - ◆ TCP6, UDP6
  - ◆ Behind NAT
  - ◆ HTTP
  - ◆ SMTP – via procmail recipes
- ◆ Completely uncentralised
- ◆ Anonymity via routing among the nodes
- ◆ Searching with distributed hash - experimental
- ◆ Rather complicated to use!
- ◆ Not very popular
- ◆ Difficult to compile the latest version – plenty of linking bug



## ◆ Multipeer

- ◆ Norwegian invention – using multicast to support download
- ◆ IPv6 compatible – written in python
- ◆ IPv6 application idea contest winner
  - ◆ <http://ipv6.hiof.no/multipeer/>

## ◆ Threedegrees

- ◆ Instant messaging + p2p stream sharing - Not really file sharing
- ◆ Protocol not known!
- ◆ Windows only



threedegrees

[www.threedegrees.com](http://www.threedegrees.com)



## *Conculsions and Questions?*

There are p2p filesharing tools with IPv6 support, however they are not the pupolar one (except Bittorrent).

They are in various support and working level

More information about IPv6 Bittorrent :  
<http://6net.niif.hu/index.php?mn=3&lg=en>

János Mohácsi  
<mohacsi@niif.hu>