Content Availability, Pollution, and Poisoning in File Sharing Peer-to-Peer Networks

Nicolas Christin
SIMS, UC Berkeley
christin@sims.berkeley.edu

Andreas S. Weigend
Weigend Associates LLC
andreas@weigend.com

John Chuang
SIMS, UC Berkeley
chuang@sims.berkeley.edu

Background

- Several petabytes of content present at any time in file sharing networks, but…
- Vast amounts of useless files (Liang *et al.*, 2005)
  - Poorly encoded or corrupted
  - Incorrect or misleading metadata
  - …
- Signal-to-noise ratio can be extremely low…

Can we rely on injecting useless content to impact usage of file sharing networks?

Motivation

- Possible defense mechanism against copyright infringement in P2P networks
  - Some companies specialize in injection of noise
    - Overpeer, Retspan, Macrovision…
- Viable technological alternative to legal recourse?
  - Difficult to prosecute individual users
- Injection of useless content does *not* require monitoring, or intrusion
  - Probably much more acceptable than most other interdiction methods in the eye of the general public
  - Does not require new “safe harbor” laws (H.R. 5211)

Related work

- Bird's eye view of network measurements
  - Effect on backbone (Sen and Wang, 2002)
  - Prevalence of P2P traffic (Saroiu *et al.*, 2002)
  - Traffic not decreasing (Karagiannis *et al.*, 2004)

- Topological properties of P2P file sharing networks
  - Gnutella (Loo *et al.*, 2003)
  - KaZaA (Liang *et al.*, 2004)
  - eDonkey (Tutschku, 2004, Le Fessant *et al*., 2004)
  - …

- Works on pollution/poisoning still rare
  - Quantification of the phenomenon (Liang *et al.*, 2005)
  - Theoretical studies of potential attacks on P2P networks (Castro *et al.*, 2002, Dumitriu *et al.*, 2005)

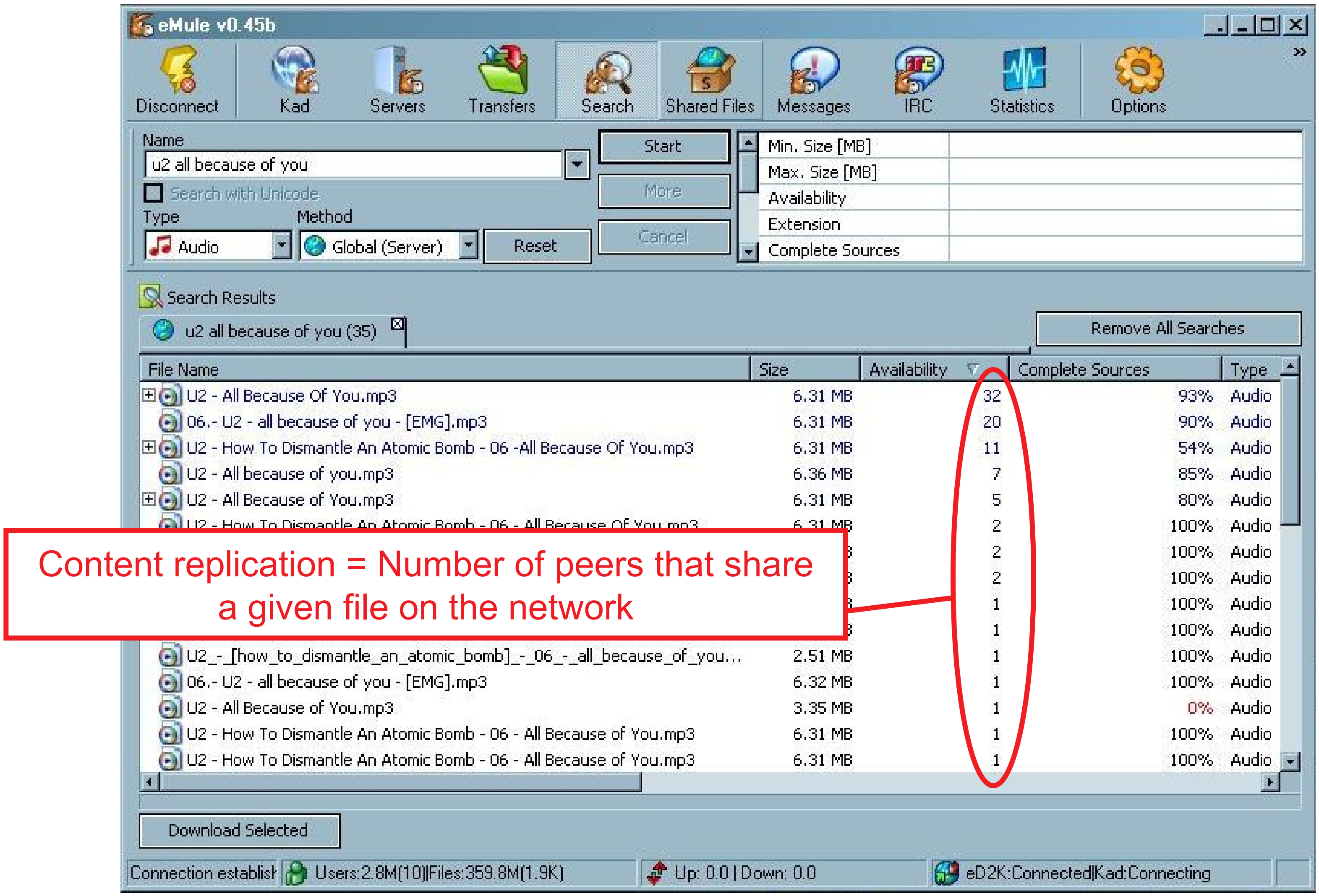
Pollution vs. Poisoning

- Network pollution
  - *Accidental* injection of unusable or low quality files
    - Happens with most (all?) content
    - Truncated, poorly encoded, …
    - Difficulties in properly "ripping" content
- Item poisoning
  - *Deliberate* injection of decoys to render usable files hard to find
    - Targets specific content
    - e.g., "American Life" by Madonna
  - Currently most popular interdiction technique

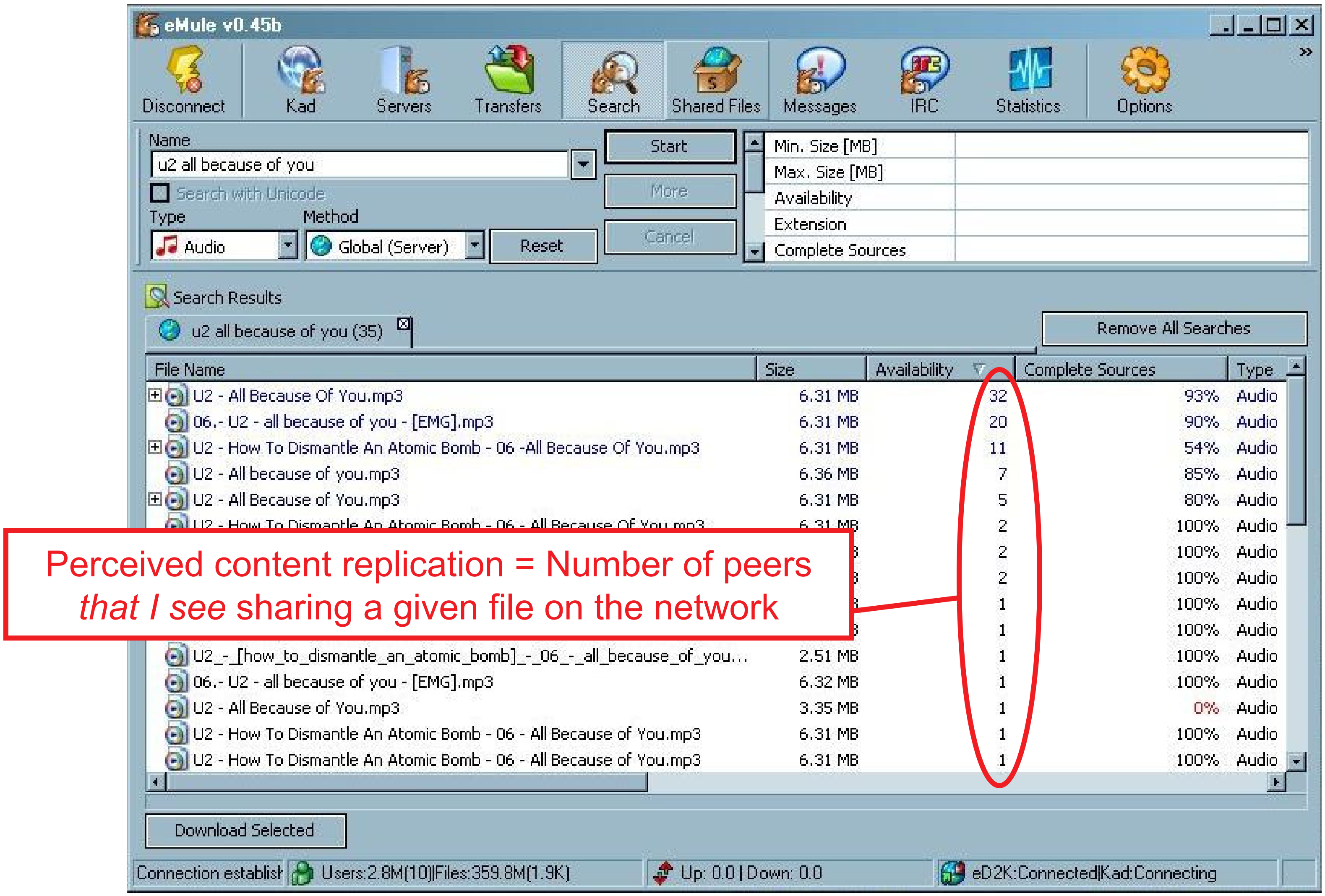
Research questions

- Above which level does pollution pose serious problems?
- Which (if any) poisoning techniques are effective?
  - Flooding?
  - More elaborate techniques?
- We'll look at the most popular P2P networks
  - FastTrack (KaZaA), eDonkey, Overnet, Gnutella
  - not BitTorrent – does not have built-in search mechanism (yet)

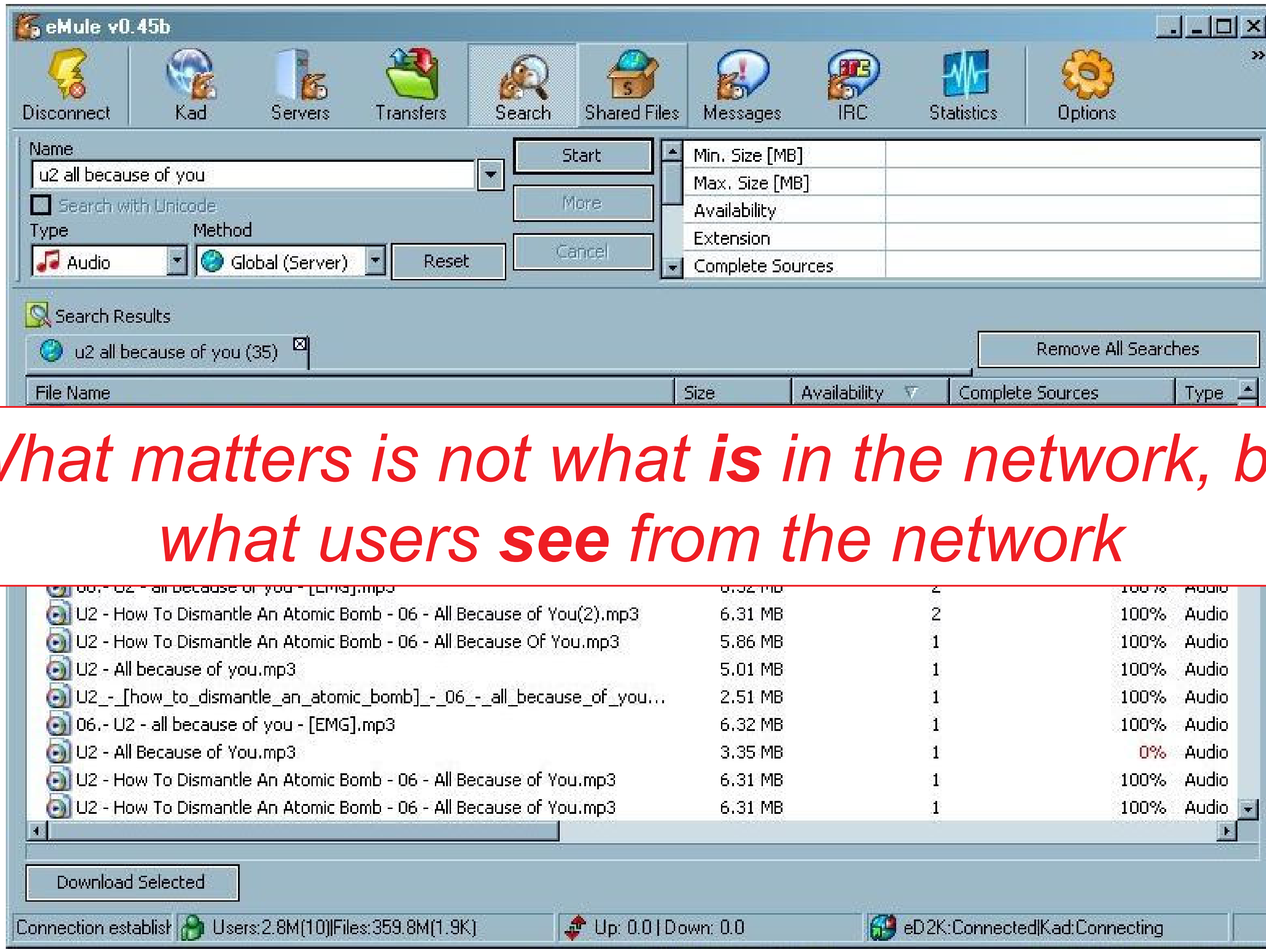
Availability vs. perceived availability



Availability vs. perceived availability



Availability vs. perceived availability



*What matters is not what **is** in the network, but what users **see** from the network*

Differing perceptions of content

- Ideally all P2P nodes should have same view of content available on the network
- In practice, different nodes have very different perceptions of content availability
  - Peers coming and going $\Longrightarrow$ Content volatility
  - Size of the network/decentralized nature imposes fish-eye view
- User view of the network conditioned by query returns
- Query returns highly dependent on P2P network topology

P2P topologies

- Most modern P2P networks use 2-level hierarchical structure
  - Leaf nodes
  - Hubs (a.k.a. supernodes, ultrapeers, servers)
    - Higher processing power, link capacity, longer uptime…
    - Act as a centralized index for a number of leaf nodes

- Exception: Overnet
  - Distributed Hash Table (all peers are equal)
  - However, Overnet clients are also part of the eDonkey network

Differences in topological structures

	eDonkey	FastTrack	Gnutella
# of nodes	≈ 2,800,000	≈ 2,500,000	≈ 1,000,000
# of hubs	40—90	25,000—40,000	10,000—100,000
Fraction of hubs	≈ 0.00002	≈ 0.015	≈ 0.05
Avg. leaf-hub connection lifetime	≈ 24 hours	≈ 30 minutes	≈ 90 minutes
Leaf promotion	Voluntary	Election	Election

Differences in topological structures

	eDonkey	FastTrack	Gnutella
# of nodes	≈ 2,800,000	≈ [illegible]	[illegible],000,000
# of hubs	40—90	25,0[illegible]—[illegible]0,000	[illegible]0—100,000
Fraction of hubs	≈ 0.00002	≈ 0.015	≈ 0.05
Avg. leaf-hub connection lifetime	≈ 24 hours	≈ 30 minutes	≈ 90 minutes
Leaf promotion	Voluntary	Election	Election

Semi- centralized network

Hubs are much more stable

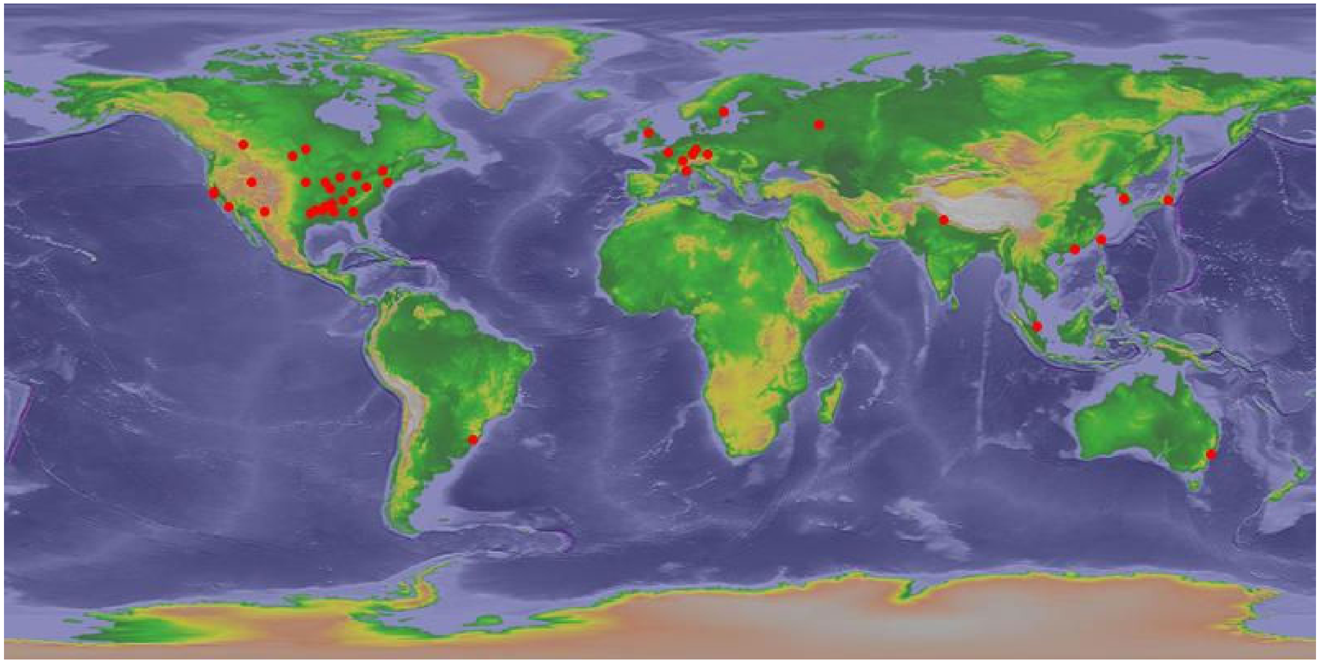
Methodology

- Perception of availability depends on time and origin of a query
  - Need to measure from different vantage points and at different times

1. Measure content availability *in absence* of poisoning
2. Evaluate effect of pollution and poisoning on measured data by numeric simulation

Measurement infrastructure

- giFT-FastTrack and MLDonkey clients
  - Linux console (text-based) applications
  - Allows for scripting
- Easy to run large scale experiments
  - 50 host machines over 18 different countries (PlanetLab)



Active measurements

- Present network with input (queries)
  - 6 movies, 6 songs, 3 software titles
  - Specialized queries (e.g., "filetype = MP3") whenever possible
  - Content not subject to any (noticeable) ongoing poisoning attack
  - Each query is issued every half-hour for 36 hours
  - For each of the four P2P networks considered, each query is sent from at least six machines

Summary of measurements w/o poisoning

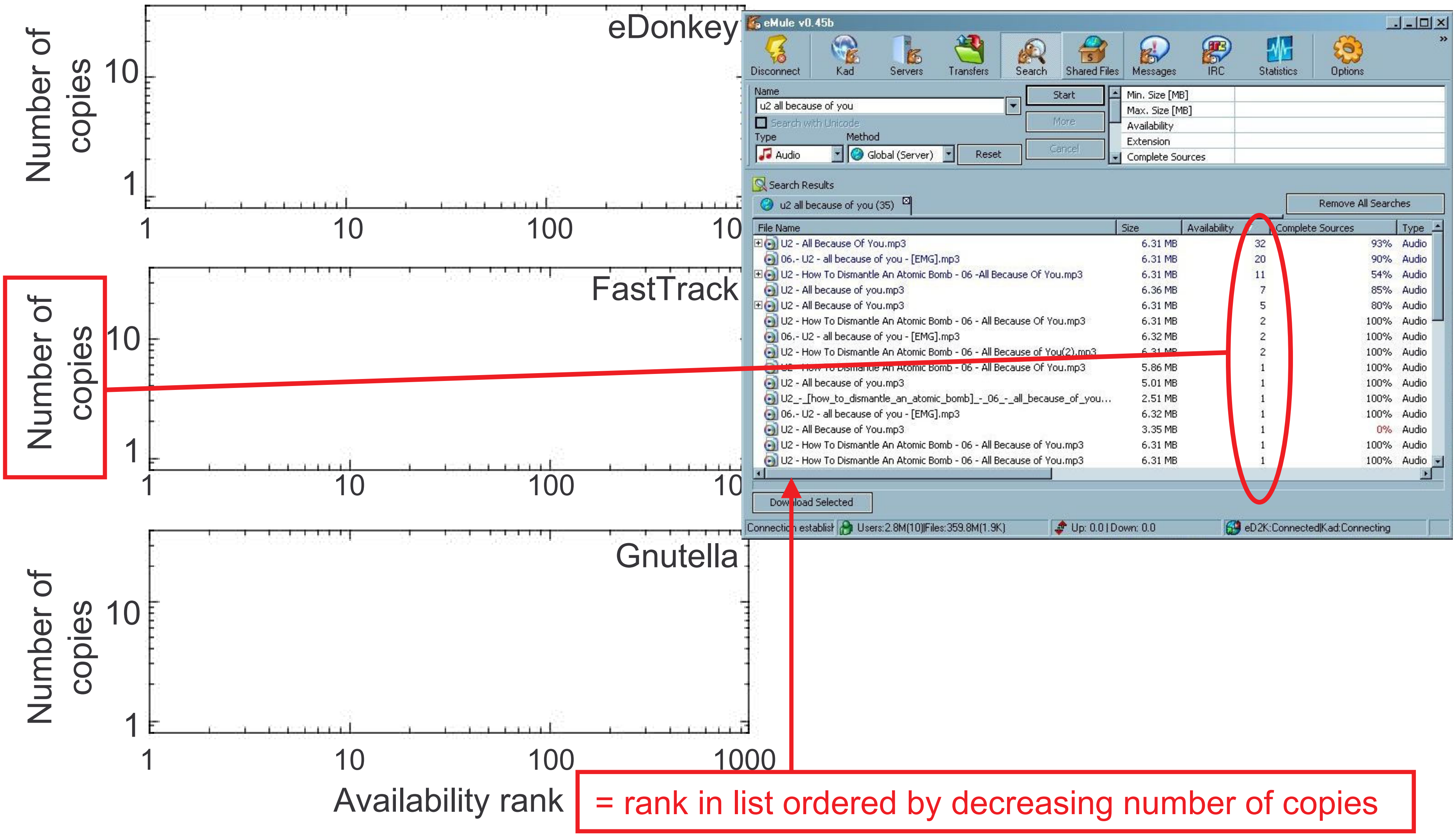
- Semi-centralized topologies (eDonkey)
  - Content remains present in the network for a while
  - Faster responses to queries
- FastTrack and Gnutella
  - Relatively low content stability
    - content comes and goes frequently
  - Apparently high levels of pollution
    - even when no poisoning
  - Manage to only download a few files
  - Confirms findings of (Liang *et al.*, 2005)

Effects of pollution

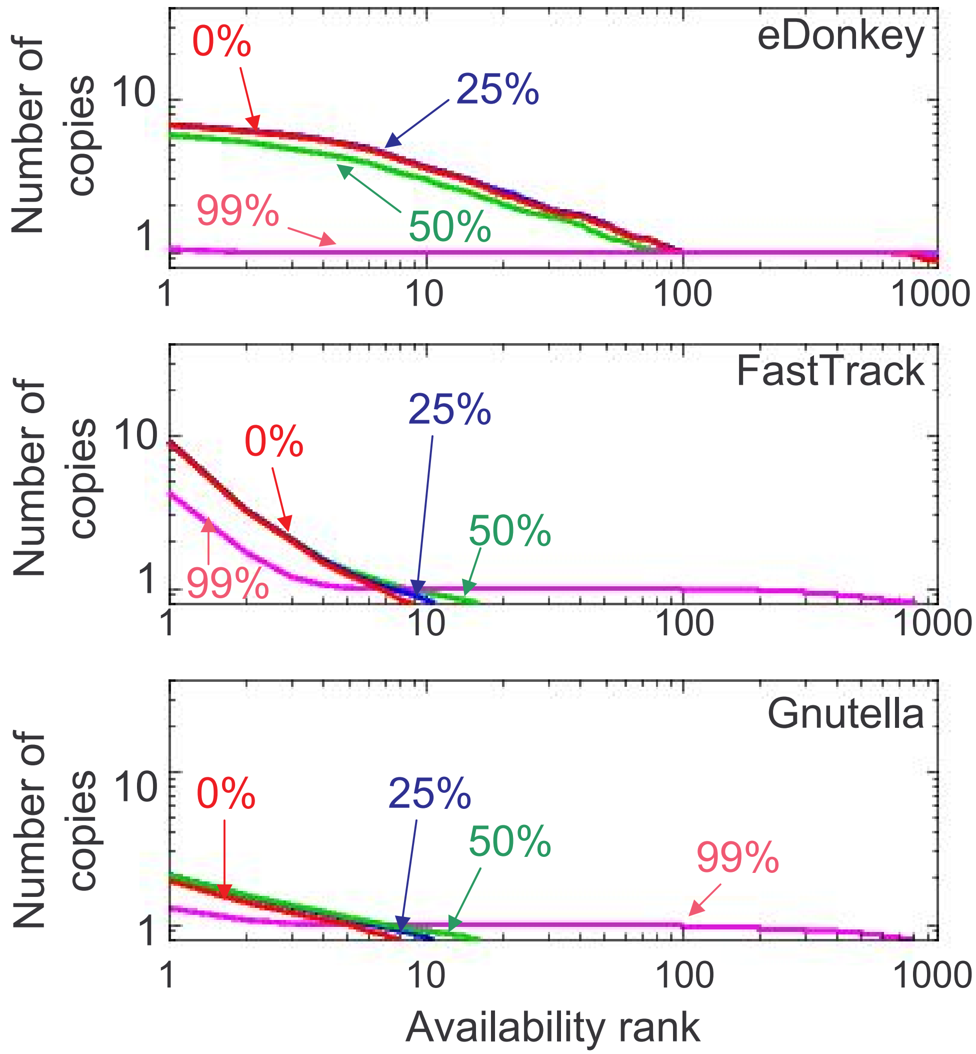
- Pollution modeled as injection of random noise in the system
  - Make $x\%$ of the query returns (uniformly) random for each measurement sample
  - Neglects propagation effects of polluted content

- Simplest poisoning technique (flooding) is nothing more than pollution at high levels
  - Should not, *in theory*, reduce availability of useful files

Pollution and perceived availability

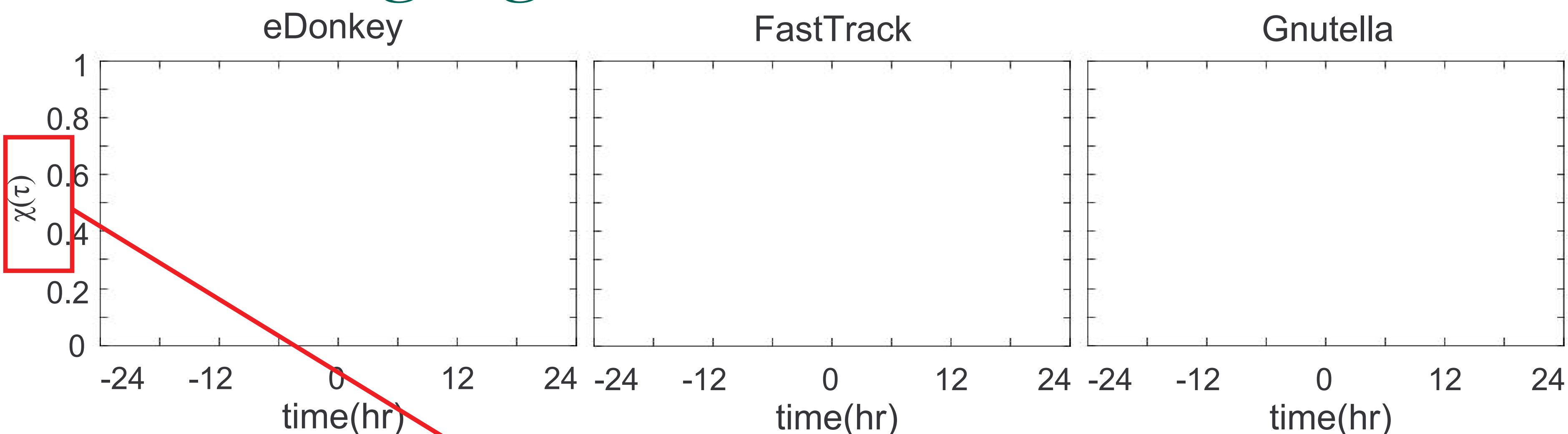


Pollution and perceived availability



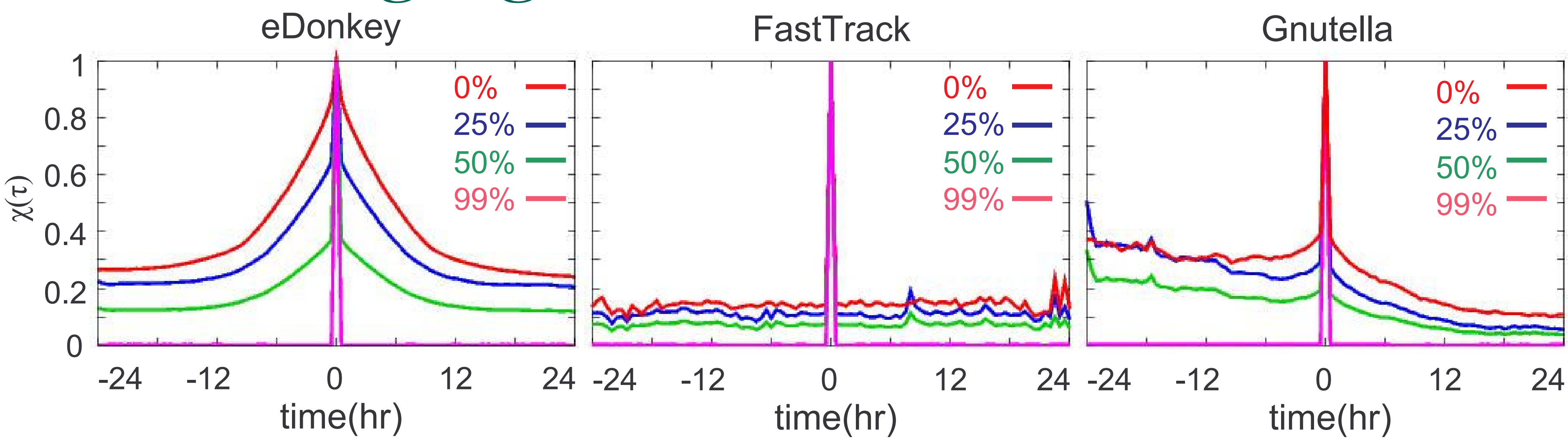
- Pollution only harmful at (very) high levels
- Decoys *may* drive usable files out of the query returns
  - Number of query returns is limited
    - FastTrack example:
      - At most 200 returns for a given query
      - No more than 5 queries in a row
- Poisoning by flooding not particularly efficient
  - e.g., need to insert 99 times as many decoys as existing files
  - … at each hub

Flooding signature



χ(τ): average probability (over all times, all clients) that an item (specific file) returned at a given time *T* is also returned at time *T+τ*

Flooding signature



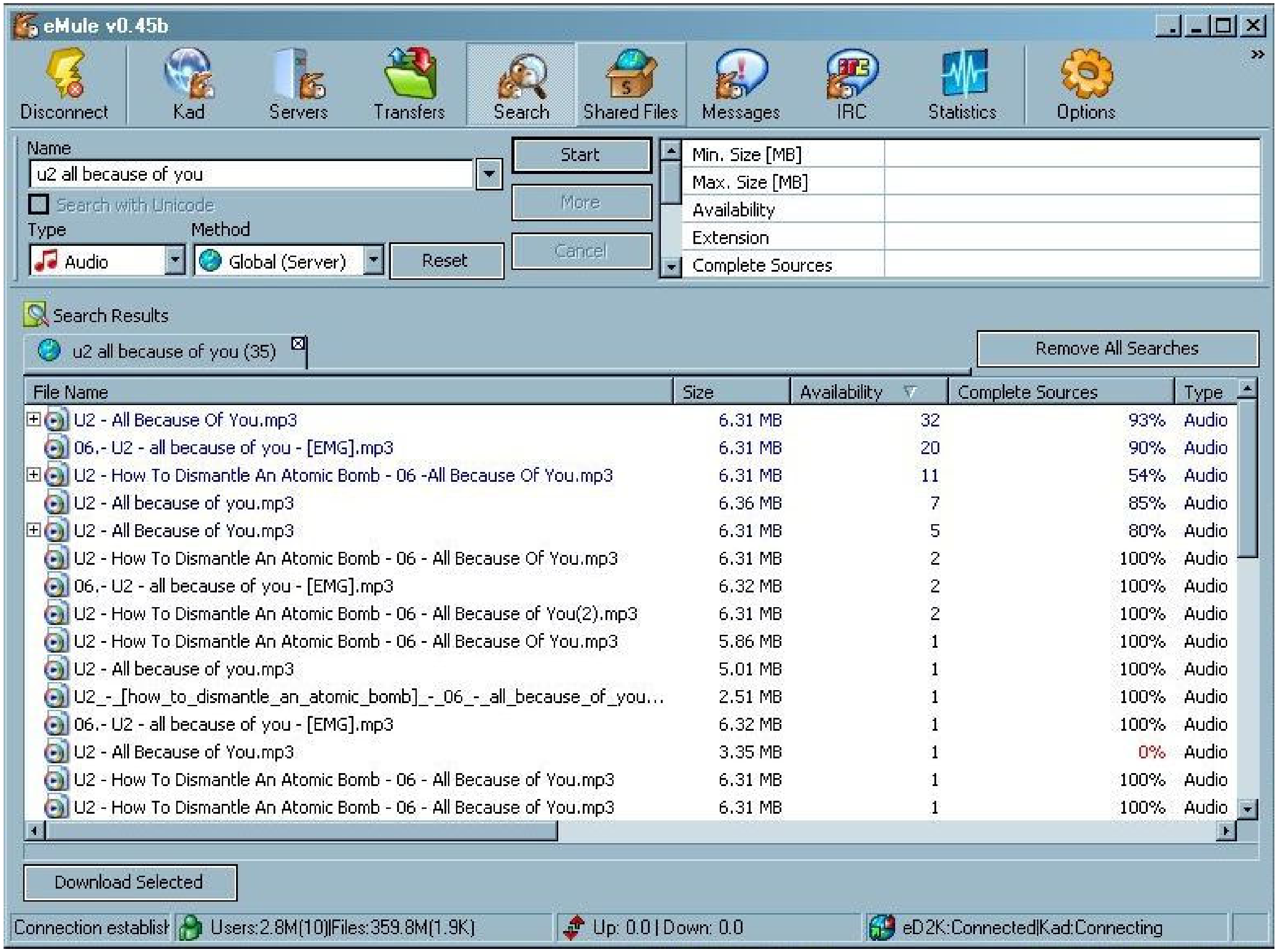
- High-levels of pollution (or poisoning by flooding) completely destroys temporal stability
- Flooding attack easy to thwart by giving precedence to items that have been seen in the network for some time

Alternatives to flooding

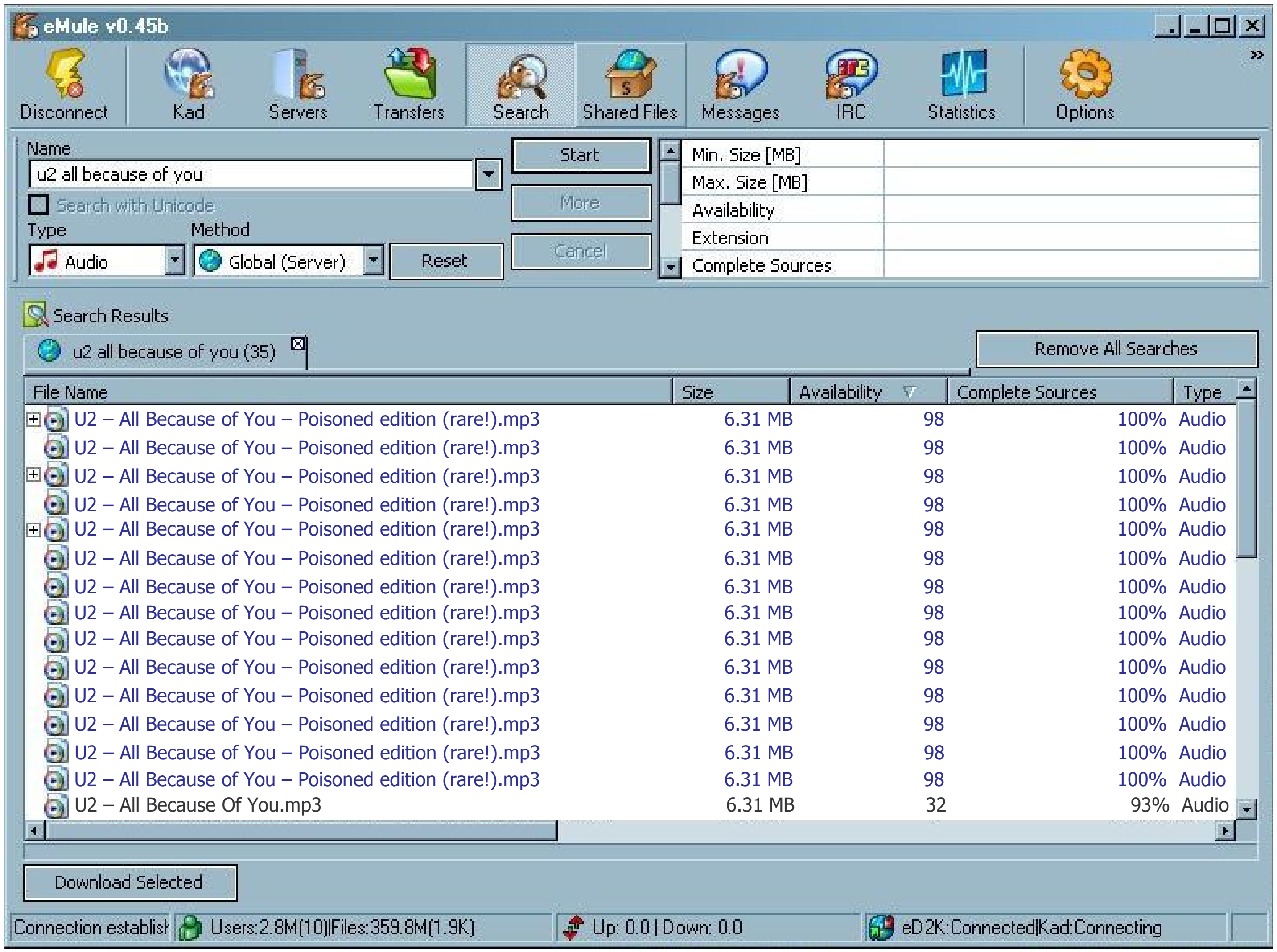
- More advanced poisoning techniques can be much less expensive and more efficient than flooding
  - A (rather detailed) list of attacks is available in a patent application from Macrovision
    - Discussed at http://mvsn-patent-app.notlong.com

  - Chunk corruption
  - Malicious routing
  - Skewing perceived availability to bias users towards downloading useless content
  - …

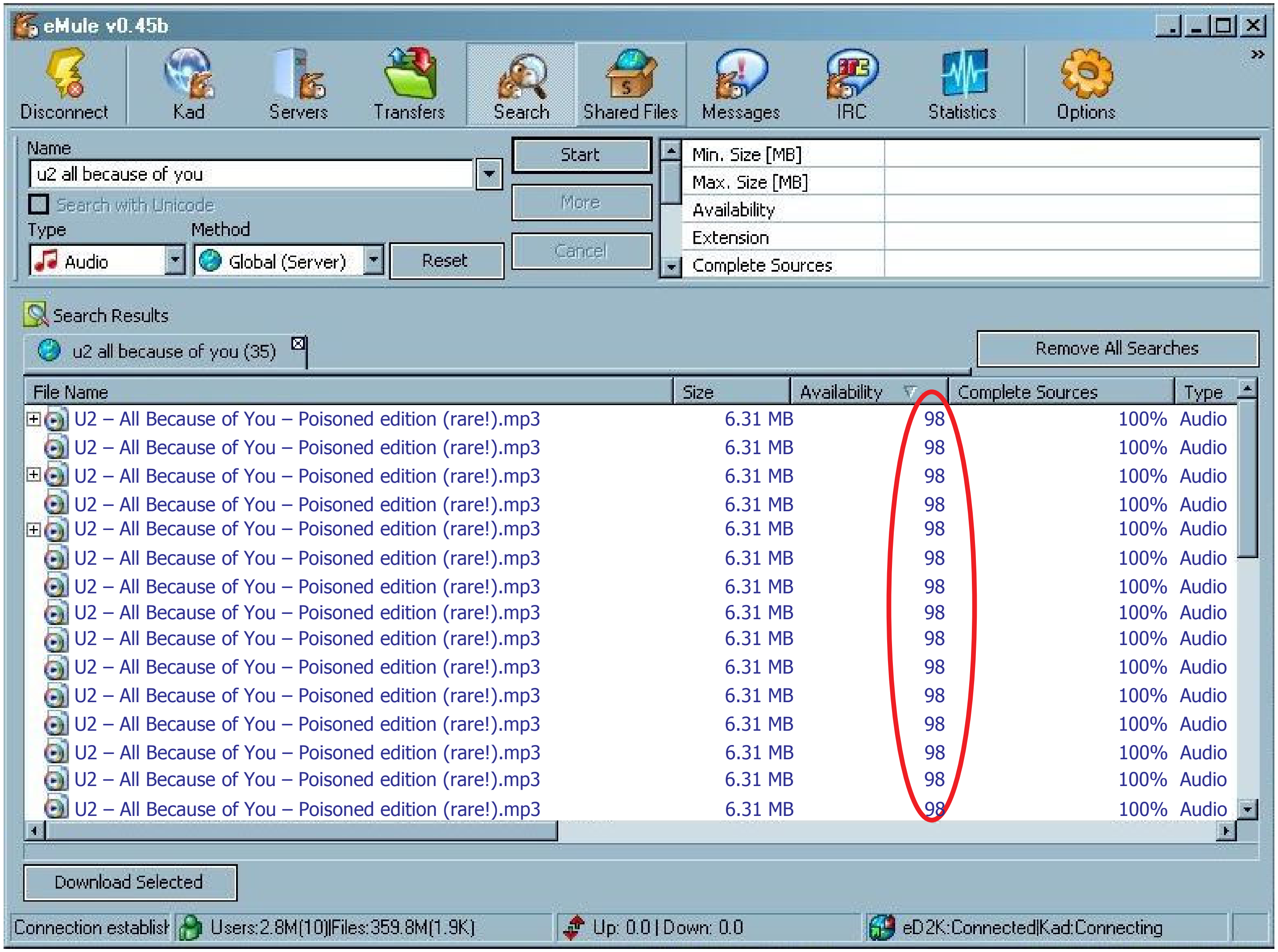
Targeting perceived availability



Targeting perceived availability



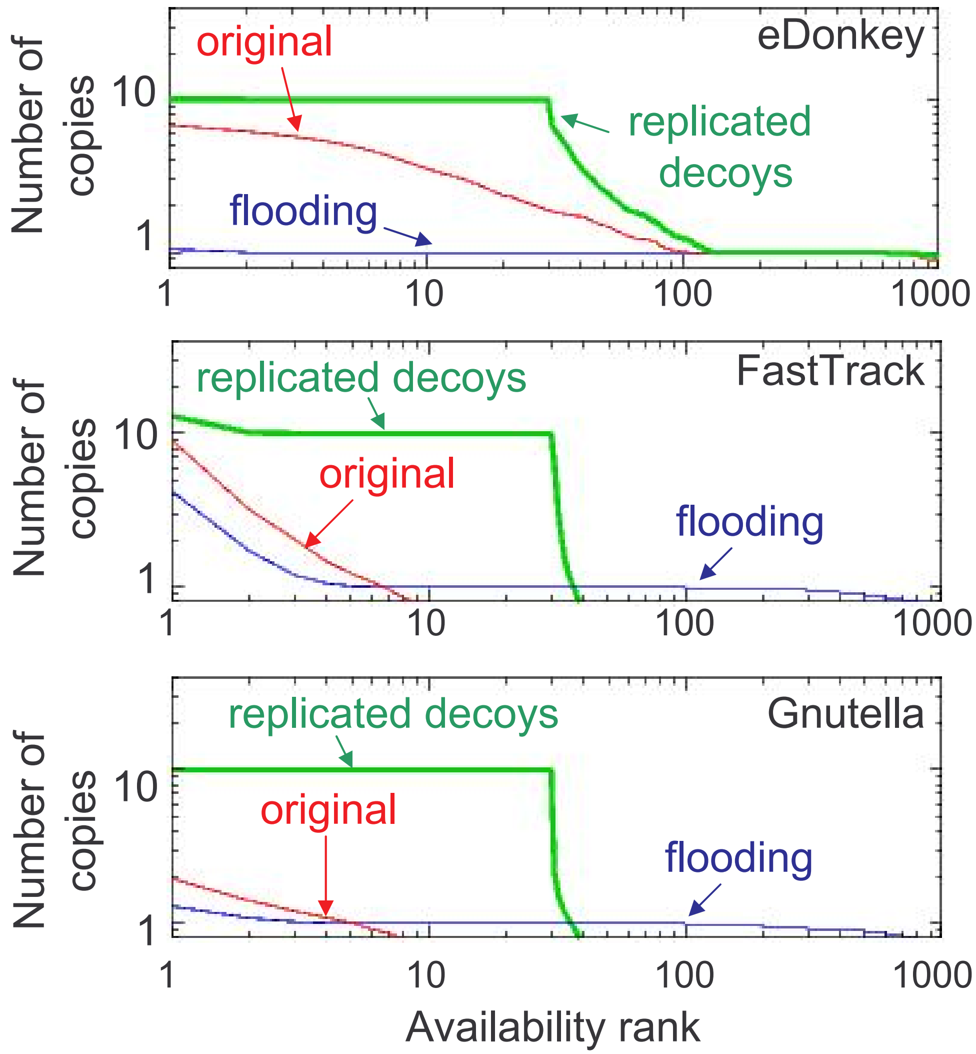
Targeting perceived availability



Targeting perceived availability

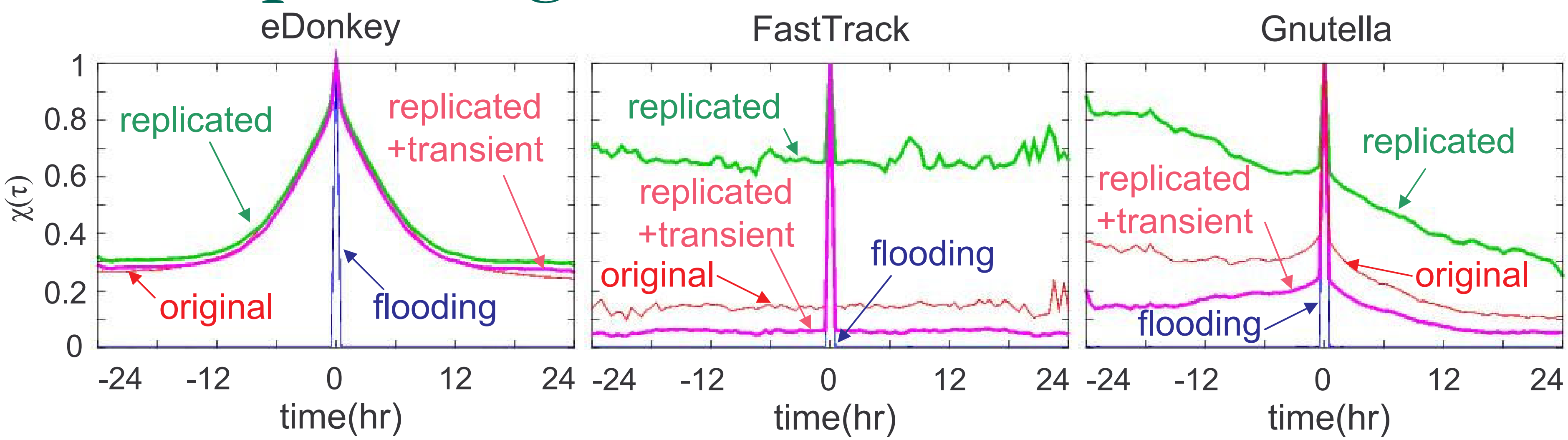
- Inject a few highly replicated decoys rather than random files
- Can in addition make replicated decoys harder to detect by frequently changing them (transient decoys)

Replicated decoy injection



- Insert 30 decoys with the same number of copies as most replicated file
- Drives useful files out of the picture
- Here only requires about 300 decoys
  - as opposed to ~9900 for flooding

Temporal signatures



- Using permanent replicated decoys leaves a rather obvious signature on the temporal stability
- Can be solved by frequently changing the (replicated) decoys

Poisoning antidotes

- Ranking by availability
  - Simplest technique
  - Efficient against random noise (if no propagation)

- Static reputation system
  - “File X is useless,” “IP address Y injects useless content”
  - Needs manual input, far from comprehensive
  - http://www.jugle.net, http://bitzi.com
- Dynamic ((semi-)automated) reputation system
  - Weighs reputation of a file as a number of factors
    - Manual input
    - Time present in the system
  - Semi-automate ban of poisoning sources
  - Unlikely such systems are *currently* deployed

Antidotes and their effectiveness

	Pollution	Flooding	Replicated decoys	Replicated, transient decoys
Ranking by number of replicas found	Yes	Somewhat	No	No
Static reputation	Somewhat	No	Yes	No
Dynamic reputation	Somewhat	Somewhat	Yes	Somewhat

The poisoning arms race

P2P designers

- Need to use several antidotes in conjunction
  - e.g., ranking by number of replicas with reputation
- Efficiency of reputation systems improved by looking at statistical characteristics
  - Temporal stability signatures

Copyright holders

- Brute force never a bad choice
  - Can be devastating if used with proper (combination of) strategies
- Clever techniques can use the reputation system to catalyze poisoning
  - False positives
  - False negatives

Summary

- Network topology plays a crucial role in how users perceive content
  - (Semi-)centralized topologies provide more stable content
- Easy to combat (involuntary) pollution
  - E.g., ranking results by number of replica found
- More advanced poisoning strategies harder to thwart
  - Arms race between poisoning techniques and reputation systems

Conclusion

Can we rely on injecting useless content to impact usage of file sharing networks?

It is far from impossible…

... and it avoids putting anyone in jail!