Security Issues and Perspectives in P2P Systems: from Gnutella to BitTorrent

Prof. Marinho P. Barcellos
marinho@acm.org
Unisinos  PUCRS

53nd Meeting of the IFIP 10.4 Working Group on Dependable Computing and Fault Tolerance

Natal, Feb. 2008
1. P2P Systems
2. Security Issues on P2P
3. Flexible P2P Security Layer
4. Pollution Control
5. On the Security of BitTorrent
6. Experiments with BitTorrent
7. Final Remarks
A broad term: there is no consensual definition

One is: “totally decentralised system where peers are all equals”, but this excludes superpeer-based schemes

Another is “class of applications that take advantage of resources at the edge of the Internet”, but this includes SETI@home
Distributed system consisting of interconnected, potentially autonomous nodes

that share resources (such as content, CPU cycles, storage and bandwidth) and

are organised in overlays networks

capable of adapting to transient populations while maintaining acceptable connectivity and performance,

typically without relying on a global central entity.
P2P main application categories

- File Sharing
- Overlay multicast (or application-level multicast)
- User collaboration and communication
- Distributed computing (P2P meets the Grid)
Summary

1. P2P Systems
2. Security Issues on P2P
3. Flexible P2P Security Layer
4. Pollution Control
5. On the Security of BitTorrent
6. Experiments with BitTorrent
7. Final Remarks
A range of security concerns for P2P

- Ensuring the *availability* of data/services provided by peers
- Verifying the *authenticity* of users/peers
- Maintain the *confidentiality* of information stored/exchanged
- Check on *integrity* of data provided by peers
- Provide *authorisation* of peers to access data or use services
- Employ *trust* & *reputation* to tell good from bad peers/data
- Ensure *anonymity* in providing, searching and using data or services
- Ensure *deniability* of provided data or services
- *Non-repudiation* of provided data or services
Attacks to authenticity

- **Sybil**: falsifying the identity of peers to create multiple, fake identities, to be used concurrently.
- If a single malicious peer can assume multiple ids:
  - In replication-based systems multiple replicas can become under control of a single malicious peer.
  - So security and reliability could not be guaranteed.
According to Doceur, in a large-scale distributed system it is not pragmatically possible to prevent a malicious peer from obtaining multiple ids.

Unless a Certification Authority or other centralised entity is employed.

However, this is clearly undesirable in a large-scale system like the P2P ones discussed here.
Attacks to availability

- **Noncooperative peer**: responds to queries, but remains silent when service or resource is requested.
- **Conventional DoS**: attack to specific nodes that provide some service or possess a given resource.
- **Excessive churn**: accelerated arrival and departure of malicious peers (more effective when replicas are used).
- **Unsolicited messages**: malicious peer engineers situations when an unsolicited reply message is sent.
- **Slow peer**: messages can be modified so that slow honest peer seems wrongly resourceful.
Attacks to availability

★ Three categories of routing attacks:
- incorrect routing of queries
- incorrect update of routes
- overlay partitioning (at bootstrap)

★ Eclipsing attack:
- malicious user has a great number of peers and coordinates them to isolate one or more honest peers
- attack can be made feasible with Sybil’s (or else by manipulating the routing tables)
Objects can be corrupted by malicious peers

Standard cryptography schemes can be used against it

A common integrity attack in P2P is related to pollution:

- *file-targeted DoS attack*: malicious peer announces one or more copies of a corrupted version of a content

- *false attack reply*: malicious peer intercepts reply from a query and announces itself as having the resource; if selected, peer sends corrupted copy
Attacks to trust and reputation

- **Whitewashing**: peer misbehaves, gets a bad reputation, but then leaves system and returns with a new id - policy towards newcomers?
- **Collusion**: peers work together to perform an attack, like providing false testimony towards malicious or honest peers
- **Traitor**: peer gathers good reputation and then misuses it
Area is vast
Collected 500+ papers (references)
On P2P and/or security (most are “and” in some degree)
Papers with tags such as “trust”, “simulation”
Publicly available - http://www.citeulike.org/user/p2p-sec
Summary

1. P2P Systems
2. Security Issues on P2P
3. Flexible P2P Security Layer
4. Pollution Control
5. On the Security of BitTorrent
6. Experiments with BitTorrent
7. Final Remarks
Context and motivation

★ The diversity of P2P applications is still small, specially in the enterprise

★ One reason is the lack of security associated with P2P

★ Difficult to develop P2P applications that address a combination of aspects of security

★ Desired:

✦ Integration of security into user application and keep it isolated from application code

✦ Allow application to establish specific and asymmetric requirements for different peers

✦ Allow gradual deployment
P2PSL

★ P2PSL is a security layer that is used along with a P2P communication middleware

★ Security modules are combined like “Lego pieces” to provide flexible, asymmetric security

★ Each peer maintains security “profiles” with mutually-exclusive sets of peers that it knows
Flexible P2P Security Layer

P2PSL

Profile C
Outgoing: Authentication + Confidentiality
Incoming: Authentication + Auditing

Profile D
Outgoing: Authentication
Incoming: Authentication + Auditing

Profile C: Peer1, Peer2
Profile D: Peer4

Profile A
Outgoing: Authentication
Incoming: Authentication
Profile A: Peer1, Peer4

Profile B
Outgoing: Authentication
Incoming: Authentication + Confidentiality
Profile B: Peer3

Authentication (e.g. digital signing)
Confidentiality (e.g. cryptography)
Authorization (e.g. policies checking)
Auditing (e.g. log generation)
Flexible P2P Security Layer

P2PSL

- Configuration
- Cryptography Module
- Digital Signature Module
- Communication Middleware

Profile B (Peer3)

Peer 2

P2PSL
Flexible P2P Security Layer

Implementation

- JXTA-based prototype implementation
- Experiments using OurGrid and syntectic application
- For more info:
Summary

1. P2P Systems
2. Security Issues on P2P
3. Flexible P2P Security Layer
4. Pollution Control
5. On the Security of BitTorrent
6. Experiments with BitTorrent
7. Final Remarks
File sharing seems to be the P2P killer application

One major associated problem is “content pollution”

“False” versions of a content, either wrongly identified or with corrupted data, are published by malicious peers to elude users into downloading undesired content.
Pollution Control

Proposed strategy

- Limit the download rate of a content version according to its reputation
- Peers download content and may vote against/for its integrity in regards to content metainfo
- If initial reputation value is low, then fewer downloads will be allowed; thereafter reputation of a correct content will grow and so the rate
- Unlike other approaches, ours mitigates the impact of a pollution attack whereby malicious peers generate a large number of polluted versions of the same content soon after the correct version has been published
Distributed schemes

- **Centralised**: a DM controls one content version (the smallest sharing unit)
- **Super-peers**: multiple DMs on super-peers networks
- **Chord-based**: using a segmented ring with DMs
- **Fully-descentralized**: no DMs, requires flooding and is more vulnerable due to consolidation

A download manager (DM) is an entity that grants downloads, collects votes and works out the reputation index of a single content version.
Evaluation

★ Scenarios:

1. non-polluted content with honest peers only - measures unnecessary slowdown when control is present but not required
2. polluted content (seeded by malicious peers) and with honest leechers - measures efficiency of the pollution control
3. non-polluted content (seeded by honest peers) with colluding peers - measures the impact of collusion attacks that produce false votes to slow down dissemination
4. polluted content (seeded by malicious peers) helped by colluding peers - measures impact of collusion attacks that produce false votes to increase dissemination

★ Metrics is number of completed downloads through time
Pollution Control

Results

Polluted content under different mechanisms
Pollution Control

Results when two attacks are combined

- Overhead caused by collusion
- Non-polluted content with colluding peers under different mechanisms

Graph showing the number of downloads over time with different control mechanisms.

Legend:
- No Control
- Centralized
- Chord
- Decentralized
Summary

1. P2P Systems
2. Security Issues on P2P
3. Flexible P2P Security Layer
4. Pollution Control
5. On the Security of BitTorrent
6. Experiments with BitTorrent
7. Final Remarks
On the Security of BitTorrent

BitTorrent

★ De facto standard for file sharing
★ Accounts for a large portion of Internet traffic
★ Many protocol implementations available through user agents (“clients”)
★ Key design aspects (with scalability and security implications):
  ✦ Network of networks: “swarms” or “torrents” (per content)
  ✦ Separation between searching and sharing
On the Security of BitTorrent

BitTorrent main elements

- .torrent file
- Tracker
- Seeders
- Leechers
- Swarm
- Pieces
- Blocks
- Bitfields
Connection policies:
- peer obtains IPs of remote peers from the tracker, and
- peer originates a number of connections to other peers, and also receives connection requests

Tit-for-tat: incentive mechanism to increase collaboration

Local Rarest First (LRF): peer selects for download the piece that is rarest to it (not necessarily rarest in the swarm)
On the Security of BitTorrent

BitTorrent state diagrams

START

WAIT_TRKCONNECT

WAIT_PEERLIST

MANAGING_PEERLIST

REC_WAIT_BITFIELD

FINISH

P2Psec

Prof. Marinho P. Barcellos - P2P-SeC

Security Issues on P2P Systems 32

53nd IFIP 10.4 WG - Natal, Feb 2008
On the Security of BitTorrent

BitTorrent Protocol: time diagram

Leecher1 -- Seed -- Leecher2 -- Tracker

- Get
- PeerList(null)
- PeerList(s)
- Get
- Handshake
- Bitfield
- Interested
- Handshake + Bitfield
- multiple blocks
- Request(p,b)
- Have(p)
- multiple pieces
- Piece(p,b)
- Unchoked
On the Security of BitTorrent

**Attack and Subversion Strategies**

- **Eclipse**: dominate connections to some honest peers
- **Piece lying**: pretend to have some pieces in order to make them rarer
- **Piece corruption**: send corrupted blocks to honest peers to make them discard and reload pieces

**Selfish behaviour vs. Malicious behaviour?**
One view: the former regards *fairness*, whereas the latter is related to causing *harm*
Assuming each honest peer will connect to a limited number of peers
Malicious peers connect to honest peers (and may be connected by honest peers)
Malicious peers do not provide data
The list of peers a honest peer knows about is randomly taken from trackers, and there will be malicious and honest peers
If honest peer is connected to malicious peers only, then it is Eclipsed
It makes no progress and eventually fails
Even if there are some honest peers connected, the download will take longer or even fail if peers leave
On the Security of BitTorrent

**Piece Lying Attack**

- LRF aims to equalise the number of copies of each piece in the swarm.
- Attack aims to interfere with this balance.
- Malicious peer lies about having one or more pieces, but never *unchokes* other peers.
- Many peers acting in collusion and lying about certain pieces induce the non-malicious peers to choose other pieces first.
- This tends to make certain pieces *artificially rarer*.
- When pieces become less common, it may slow down the swarm or even cause a swarm failure.
On the Security of BitTorrent

**Piece Corruption Attack**

- It is not possible for a peer to find out the set of corrupted blocks in a piece
  - If a piece is 4MB, a single corrupted block (16KB) causes the whole piece (256 blocks in this case) to be discarded
- Some agents have primitive mechanisms to defend against this attack ("IP filters")
- Basic attack approach:
  - Unchokes honest peer
  - Waits for request
  - Send block with random content
  - Choke honest peer or remain silent
Attack impact evaluation

★ Evaluation through simulation (more on this later)
★ Interested on the initial stage of a swarm: 100 peers arrive in the 10 minutes (flash crowd)
★ Content (64 MB) and bandwidth (1024 Kbps/256 Kbps)
★ Initial permanent seeder in a fair network: ratio 1.0
★ Monitor changes in peer population (seeders, leechers) and completed downloads
★ Under different kinds of attacks
★ Parameter sweep
On the Security of BitTorrent

Attack impact evaluation

Impact of Eclipsing attacks for different number of Sybils

some peers fail to complete
On the Security of BitTorrent

**Countermeasures?**

- We have defined a peer rotation algorithm that defends against Eclipse and Piece Lying attack.
- And a reputation-based malicious peer detection and blacklisting.
- Results will soon appear.
- For impact evaluation, see IEEE P2P 2007.
1. P2P Systems
2. Security Issues on P2P
3. Flexible P2P Security Layer
4. Pollution Control
5. On the Security of BitTorrent
6. Experiments with BitTorrent
7. Final Remarks
Experiments with BitTorrent

★ Challenges for simulation:
  ✦ Complex, engineered protocol requires rich model
  ✦ Large-scale scenarios require potentially lengthy simulations

★ Challenges for experimental evaluation:
  ✦ Large-scale distributed system very hard to set up and monitor
  ✦ Uncontrolled environment prevents reproducibility

★ TorrentLab: integrated environment for BitTorrent evaluation
  ✦ Simulation campaigns
  ✦ Experimental evaluation
TorrentLab Architecture

- Modeling environment (scenario description)
- Cluster or grid middleware (distributed infrastructure)
- Simulation environment: TorrentSim
- Experimental environment: TorrentExp
- Result consolidation environment
TorrentSim

Experiments with BitTorrent
Experiments with BitTorrent

Validation of TorrentLab

- Comparison between results produced by simulation and experimental evaluation

  *We are in the process of...*

- Comparing TorrentLab results with analytical evaluation studies and anecdotal evidence

- Conducting live experiments in Wide-Area Networks & Grids, similarly to PlanetLab
Summary

1. P2P Systems
2. Security Issues on P2P
3. Flexible P2P Security Layer
4. Pollution Control
5. On the Security of BitTorrent
6. Experiments with BitTorrent
7. Final Remarks
Final Remarks

★ P2P design makes feasible large-scale distributed systems
★ Key P2P properties include descentralization, peer autonomy, transient population with potential churning
★ Hard to provide any guarantees
★ An overview of personal research efforts in the area of P2P security:
  ✦ Flexible security layer for P2P applications
  ✦ pollution control
  ✦ BitTorrent security and evaluation
  ✦ Not seen: a protocol for flexible secure service discovery
Acknowledgments

- People who have been in some degree involved with this research on P2P and security
  - Marlom Konrath, Juliano Freitas, Daniel Bauermann, Eduardo Moschetta, Giovani Facchini, Gabriel Pedebos, André Detsch
  - Rodrigo Mansilha, Rodolfo Antunes, Lucas Seewald, Carlos Schmitt, Henrique Sant’anna
  - Prof. Luciano Paschoal Gaspary, Prof. Francisco Brasileiro

- Research agencies
  - CNPq, CAPES and FAPERGS