**American Journal of Networks and Communications**
2013; 2(3): 67-72
Published online June 20, 2013 (http://www.sciencepublishinggroup.com/j/ajnc)
doi: 10.11648/j.ajnc.20130203.13

# Authority system to prevent privacy protection in Peer-to-Peer network system

**S. Uvaraj[1], N. Kannaiya Raja[2]**

[1]Arulmigu Meenakshi Amman College of Engineering, Kanchipuram
[2]Defence Engineering College, Ethiopia

**Email address:**
ujrj@rediffmail.com(S. Uvaraj), kannaiyaraju123@gmail.com(N. K. Raja)

**To cite this article:**
S. Uvaraj, N. Kannaiya Raja. Authority System to Prevent Privacy Protection in Peer-to-Peer Network System. *American Journal of Networks and Communications*. Vol. 2, No. 3, 2013, pp. 67-72. doi: 10.11648/j.ajnc.20130203.13

**Abstract:** Collusive piracy is the main source of intellectual property violations within the boundary of a P2P network. Paid clients (colluders) may illegally share copyrighted content files with unpaid clients (pirates). Such online piracy has hindered the use of open P2P networks for commercial content delivery. We proposed a proactive content poisoning scheme to stop colluders and pirates from alleged copyright infringements in P2P file sharing. The basic idea is to detect pirates timely with identity-based signatures and time-stamped tokens. The scheme stops collusive piracy without hurting legitimate P2P clients by targeting poisoning on detected violators, exclusively. We developed a new peer authorization protocol (PAP) to distinguish pirates from legitimate clients. Detected pirates will receive poisoned chunks in their repeated attempts. Pirates are thus severely penalized with no chance to download successfully in tolerable time. Based on simulation results, we find 99.9 percent prevention rate in Gnutella, KaZaA, and Freenet. We achieved 85-98 percent prevention rate on eMule, eDonkey, Morpheus, etc. The scheme is shown less effective in protecting some poison-resilient networks like BitTorrent and Azureus. Our work opens up the low-cost P2P technology for copyrighted content delivery. The advantage lies mainly in minimum delivery cost, higher content availability, and copyright compliance in exploring P2P network resources.

**Keywords:** Peer-to-Peer, Content Delivery Network, Reputation System, Colluder, Content Poisoning and Network Security

## 1. Introduction

PEER-TO-PEER (P2P) networks are most cost-effective in delivering large files to massive number of users [3]. Unfortunately, today's P2P networks are grossly abused by illegal distributions of music, games, video streams, and popular software. These abuses have not only resulted in heavy financial loss in media and content industry, but also hindered the legal commercial use of P2P technology. The main sources of illegal file sharing are peers who ignore copyright laws and collude with pirates. To solve this peer collusion problem, we propose a copyright-compliant system for legalized P2P content delivery. Our goal is to stop collusive piracy within the boundary of a P2P content delivery network. In particular, our scheme appeals to protecting large-scale perishable contents that diminish in value as time elapses. Traditional content delivery networks (CDNs) use a large number of surrogate content servers over many globally distributed WANs. The content distributors need to replicate or cache contents on many servers. The bandwidth demand and resources needed to maintain these CDNs are very expensive. A P2P content network significantly reduces the distribution cost. P2P networks improve the content availability, as any peer can serve as a content provider. P2P networks are inherently scalable, because more providers lead to faster content delivery.

We use identity-based signatures (IBS) [4] to secure file indexes. IBS offers similar level of security as PKI-based signatures with much less overhead. We apply discriminatory content poisoning against pirates. We focus on protection of decentralized P2P content networks. Protecting centralized P2P networks like Napster is much simpler than the scheme we proposed because of centralized indexing.

### *1.1. Types of Clients*

- Honest clients
- Colluders clients
- Pirates clients

Honest or legitimate clients are those that comply with the copyright law not to share contents freely. Pirates are peers attempting to download some content files without paying or authorization. The colluders are those paid clients who share the contents with pirates. Pirates and colluders coexist with the law-abiding clients. Content poisoning is implemented by deliberate falsification of the file requested by pirate.

## 2. Our Approach and Contributions

Content poisoning is often treated as a security threat to P2P networks. To our best knowledge, using selective content poisoning to prevent collusive piracy has not been explored in the past. We offer the very first proactive poisoning approach to curtailing copyright violation in P2P networks. We make the following specific contributions towards P2P content delivery.

### *2.1. Distributed Detection of Colluders and Pirates*

We develop a protocol that identifies a peer with its endpoint address. File index format is changed to incorporate a digital signature based on this identity. A peer authentication protocol is developed to establish the legitimacy of a peer when it downloads and uploads the file. Using IBS, our system enables each peer to identify unauthorized peers or pirates without the need for communication with a central authority.

### *2.2. Proactive Content Poisoning of Detected Pirates*

Our protocol requires sending poisoned chunks to any detected pirate requesting a protected file. If all clients simply deny download request without poisoning, the pirates can still accumulate clean chunks from colluders that are willing to share. With poisoning, pirates are forced to discard even clean chunks received.

### *2.3. Containment of Peer Collusion to Stage Piracy*

We recognize that peer collusion is inevitable: a paid customer may intentionally collude with pirates; a pirate may also hack into client hosts and turn them into unwilling colluders. Our system is designed so that even with large number of colluders, a pirate will still suffer from intolerably long download time. We also present a random collusion detection mechanism to further enhance our system.

### *2.4. Trusted P2P Platform for Copyrighted*

Content Delivery Hardware investment for P2P content delivery is much lower than that required in any existing CDNs. Our system only uses a few distribution agents to serve large number of clients. The system is highly scalable, robust to peer and link failures, and easily deployed. All claimed advantages are backed by performance analysis and simulation results.

## 3. Copyright-Protected P2P Networks

This section specifies the system architecture, client joining process, pirate poisoning mechanism, and colluder detection that we built in the newly proposed copyright-protection scheme.

### *3.1. Trusted P2P Network Architecture*

A protected P2P content delivery network, consisting of paid clients, colluders, pirates, and distribution agents. The design goal is to prevent pirates from downloading copyrighted files from colluders. Proactive poisoning is applied to pirates only without hurting paid clients. Only a handful of agents are used to handle the bootstrap and distribution of requested digital contents.

To join the system, clients submit the requests to a transaction server which handles purchasing and billing matters.

A private key generator (PKG) is installed to generate private keys with identity-based signatures (IBS) for securing communication among the peers. The PKG has a similar role of a certificate authority (CA) in PKI services. The difference lies in that CA generates public keys distributed in IEEE 509 certificates, while PKG takes much lower overhead to generate private keys, which are used by local hosts.

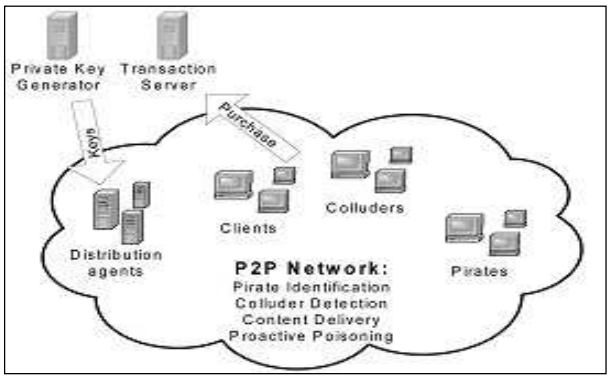


***Fig 1.*** *A secured P2P platform for copyright-protected content delivery*

This open network is accessed by a large number of paid clients, some colluders or pirates, and a few distribution agents. The system design prevents pirates from downloading copyrighted files from colluders. The transaction server and PKG are only used initially when peers are joining the P2P network. With IBS, the communication between peers does not require explicit public key, because the identity of each party is used as the public key. In our system, file distribution and copyright protection are completely distributed. Based on past experience, the number of peers sharing or requesting the

same file at any point of time is around hundreds. Depending on the variation of the swamp size, only a handful of distribution agents is needed. For example, it is sufficient to use 10 PC-based distribution agents to handle a swamp size of 2,000 peers.

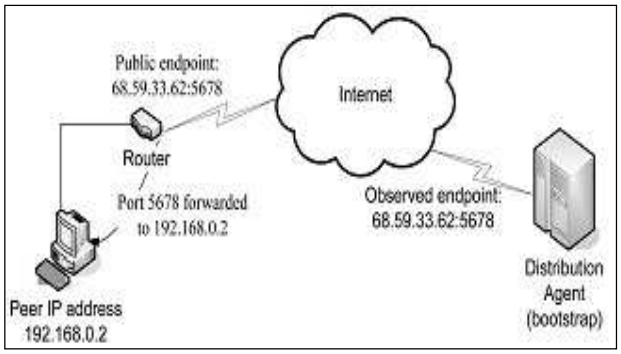


***Fig 2.*** *The bootstrap agent observes end-point address p= 68:59:33:62: 5678 in a trust-enhanced P2P network*

Fig. 2 depicts an example: A peer has an IP address 192.168.0.2 leased from its local router. It is listening to port 5,678 forwarded by the router. When communicating with the bootstrap agent, the peer announces its listening port number. The bootstrap agent calls an Observe () subroutine, which verifies that the same peer is indeed reachable via the claimed port, although its public IP address is actually 68.59.33.62. Hence, the peer is identified by 68.59.33.62:5678. The detail of Observe () is as follows: when a peer sends message to its bootstrap agent through outgoing port, agent attaches a random number (nonce) in the reply. The agent then sends a message to the advertised listening port 68.59.33.62:5678, asking the peer to send back the nonce. If the peer replies correctly, then its endpoint is verified. The endpoint address is used as peer's public key. There is no need to encrypt the file body. This reduces the system overhead. Enabling peers behind NAT without a static listening port requires a hole-punching mechanism. The system uses the bootstrap agent to forward the incoming requests. The identities of all agents, except the bootstrap agent, are hidden from clients. This stops a malicious node to blacklist or attack the distribution agents.

# 4. System Implementation

In a P2P content distribution network, only the content owner can verify the user ID/password pair; peers cannot check each other's identity. Revealing a user's identity to other peers violates his or her privacy. To solve this problem, we developed a PAP protocol. First, we apply IBS to secure file indexing. Then we outline the procedure to generate tokens. Finally, we specify the PAP protocol that authorizes file access to download by peers. Secure File Indexing In a P2P file-sharing network, a file index is used to map a file ID to a peer endpoint address. When a peer requests to download a file, it first queries the indexes that match a given file ID. Then the requester downloads from selected peers pointed by the indexes. To detect pirates from paid clients, we propose to modify file index to include three interlocking components: an authorization token, a timestamp, and a peer signature. Each legitimate client has a valid token assigned by its bootstrap agent. The timestamp indicates the time when a token expires. Thus, the peer needs to refresh the token periodically.

### *4.1. Protection in Peer Joining Process*

For a peer to join the network it first logs in to a transaction server to purchase the content after transaction, the client receives a digital receipt containing the content title, client ID, etc. This receipt is encrypted such that only content owner and distribution agent can decrypt. The client receives the address of the bootstrap agent as its point of contact. The joining client authenticates with the bootstrap agent using the digital receipt. The session key assigned by the transaction server secures their communication. Since the bootstrap agent is set up by the content owner, it decrypts the receipt and authenticates its identity. The bootstrap agent requests a private key from PKG and constructs an authorization token, accordingly. Let k be the private key of content owner and id be the identity of the content owner. We use Ek(msg) to denote the encryption of message with key k. The Sk(msg) denotes a digital signature of plaintext msg with key k. The client is identified by user ID and the file by file ID. Each legitimate peer has a valid token. The token is only valid for a short time so that a peer needs to refresh the token periodically. To ensure that peers do not share the content with pirates, the trusted P2P network modifies the file-index format to include a token and IBS peer signature. Peers use this secured file index in inquiries and download requests. Seven messages are specified below to protect the peer joining process:

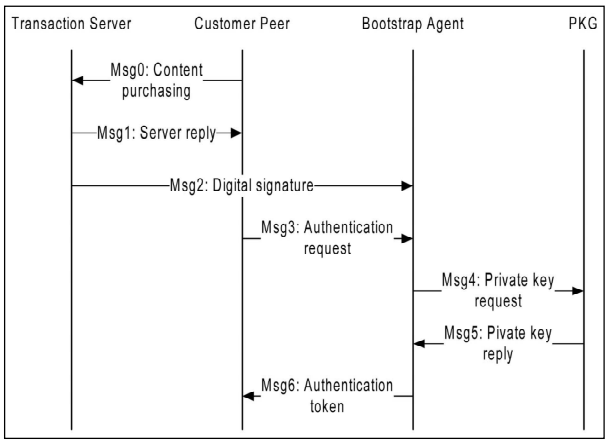


***Fig 3.*** *The protected peer joining process for copyrighted P2P content delivery. Seven messages are used to secure the communicationsamong four parties involved*

The protected peer joining process for copyrighted P2P content delivery. Seven messages are used to secure the

communications among four parties involved.

| **Seven Messages Are Specified Below To Protect The Peer Joining Process:** |
|---|
| Msg0: Content purchase request; |
| Msg1: BootstrapAgentAddress, Ek (digital_receipt, Bootstrap-Agent_session_key); |
| Msg2: Adding digital signature Ek (digital_receipt); |
| Msg3: Authentication request with userID, fileID, Ek (digital_receipt); |
| Msg4: Private key request with privateKeyRequest (observed peer address); |
| Msg5: PKG replies with privateKey; |
| Msg6: Assign the authentication token to the client. |

### *4.2. Pirate Identification*

In a P2P content distribution network, only the content owner can verify the user ID/password pair; peers cannot check each other's identity. Revealing a user's identity to other peers violates his or her privacy. To solve this problem, we are using a PAP protocol. First, we apply IBS to secure file indexing. Then we outline the procedure to generate tokens. Finally, we specify the PAP protocol that authorizes file access to download by peers.

#### *4.2.1. Secure File Indexing*

In a P2P network, a file index is used to map a file ID to a peer endpoint address. When a peer requests to download a file, it first queries the indexes that match a given file ID. Then the requester downloads from selected peers pointed by the indexes. To detect pirates from paid clients, we propose to modify file index to include three interlocking components: an authorization token, a timestamp, and a peer signature. Each legitimate client has a valid token assigned by its bootstrap agent. The timestamp indicates the time when a token expires. Thus, the peer needs to refresh the token periodically. This short-lived token is designed for protecting copyright against colluders. The cost at each distribution agent to refresh the client tokens is rather limited, as shown via experiments. The peer signature is signed with the private key generated by PKG. This signature proves the authenticity of a peer. Download requests make explicit references to file indexes. The combined effects of the three extra fields ensure that all references to the file indexes are secured. Peers identify the pirates by checking the validity of the token and the signature in a file index. These features secure the P2P network operations to safeguard the sharing of clean contents among the paid clients.

#### *4.2.2. Token Generation*

First, both the transaction server and the PKG are fully trusted. Their public keys are known to all peers. The PAP protocol consists of two integral parts: token generation and authorization verification. When a peer joins the P2P network, it first sends authorization request to the bootstrap agent. All messages between a peer and its bootstrap agent are encrypted using the session key assigned by the transaction server at purchase time.

The authorization token is generated by Algorithm 1 specified below. A token is a digital signature of a three-tuple: {peer endpoint, file ID, timestamp} signed by the private key of the content owner. Since bootstrap agent has a copy of the digital receipt sent by transaction server, verifying the receipt is thus done locally. The Decript (Receipt) function decrypts the digital receipt to identify the file L. The Observe (requestor) returns with the endpoint address p. The Owner Sign (λ; p; ts) function returns with a token. Upon receiving a private key, the bootstrap agent digitally signs the file ID, endpoint address, and timestamp to create the token. The reply message contains a four-tuple: {endpoint address, peer private key, timestamp, token}. The reply message from bootstrap agent is encrypted using the assigned session key.

**Algorithm 1. Token Generation**

Input: Digital Receipt

Output: Encrypted authorization token T

Procedures :
Step 01: if Receipt is invalid,
Step 02: deny the request;
Step 03: else
Step 04: λ = Decrypt(Receipt);
// λ is file identifier decrypted from receipt
Step 05: p = Observe(requestor);
// p is endpoint address as peer identity
Step 06: k = PrivateKeyRequest (p);
// Request a private key for user at p
Step 07: Token T = OwnerSign(f; p; ts)
// Sign the token T to access file f
Step 08: Reply = {k; p; ts;T} // Reply with key, endpoint address, timestamp, and the token
Step 09: SendtoRequestor {Encrypt(Reply)}
// Encrypt reply with the session key
Step 10: end if

### *4.3. Proactive Poisoning*

The PAP protocol is formally specified below. A client must verify the download privilege of a requesting peer before clean file chunks are shared with the requestor. If the requestor fails to present proper credentials, the client must send poisoned chunks. In PAP, a download request applies a token T, file index ø, timestamp ts, and the peer signature S. If any of the fields are missing, the download is stopped. A download client must have a valid token T and signature S. Two pieces of critical information are needed: public key K of PKG and the peer endpoint address p. Algorithm 2 verifies both token T and signature S. File index ø(λ,p) contains the peer endpoint address p and the file ID λ. Token T also contains the file index information and ts indicating the expiration time of the token. The Parse (input) extracts timestamp ts, token T, signature S, and index ø from a download request. The function Match (T; ts, K) checks the token T against public key K. Similarly, Match (S; p) grants access if S matches with p.

**Algorithm 2. Peer Authorization Protocol**

Input: T = token, ts = timestamp, S = peer signature, and

ø(λ,p) = file index for file λ at endpoint p

Output: Peer authorization status

True: authorization granted

False: authorization denied

Procedures :

Step 01: Parse (input) = {T; ts;S; ø(λ,p)}
// Check all credentials from a input request
Step 02: p = Observe(requestor);
// detect peer endpoint address p //
Step 03: if {Match (S; p) fails},
//Fake endpoint address p detected //return false;
Step 04: endif
Step 05: if {Match(T; ts;K) fails},
return false;
// Invalid or expired token detected //
Step 06: endif
Step 07: return true;

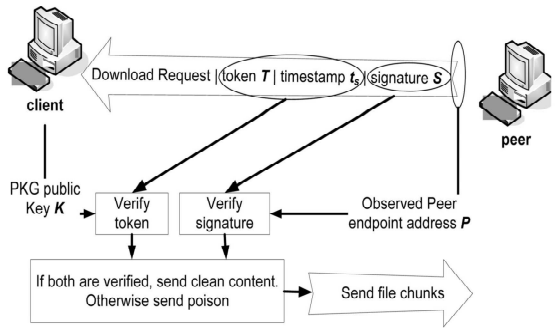


***Fig 4.*** *The PAP enables instant detection of a pirate upon submitting an illegal download request.*

### *4.4. Data Flow*

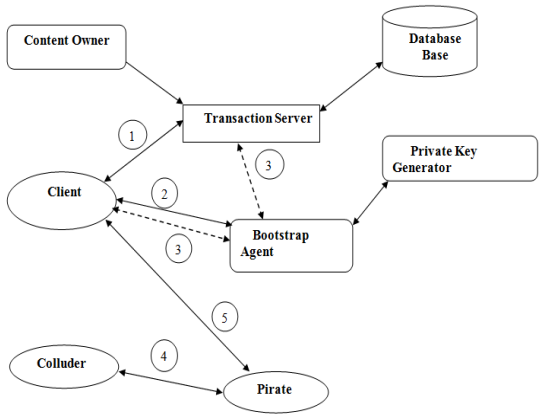


***Fig 5.*** *Dataflow diagram*

There are four types of peers coexist in the P2P network: clients (honest or legitimate peers), colluders (paid peers sharing contents with others without authorization), distribution agents (trusted peers operated by content owners for file distribution), and pirates (unpaid clients downloading content files illegally).

1. Client joining process
2. Get token from Bootstrap Agent
3. Download process
4. Pirate receive clean file chunks
5. Pirate receives poisoned chunks.

Clients are those that comply with the copyright law not to share contents freely. Pirates are peers attempting to download some content files without paying or authorization. The colluders are those paid clients who share the contents with pirates.

A private key generator (PKG) is installed to generate private keys with IBS for securing communication among the peers. The PKG has a similar role of a certificate authority (CA) in PKI services. The difference lies in the fact that CA generates the public/private key pairs, while PKG only generates the private key.

## 5. Proactive Content Poisoning

In this approach, modified file index format enables pirate detection. PAP authorizes legitimate download privileges to clients. Content distributor applies content poisoning to disrupt illegal file distribution to unpaid clients. The system is enhanced by randomized collusion detection. In our system, a content file must be downloaded fully to be useful. Such a restraint is easily achievable by compressing and encrypting the file with a trivial password that is known to every peer. This encryption does not offer any protection of the content, except to package the entire file for distribution.

Fig. 4 illustrates the proactive content poisoning mechanisms built in our enhanced P2P system. If a pirate sends download request to a distribution agent or a client, then by protocol definition, it will receive poisoned file chunks. If the download request was sent to a colluder, then it will receive clean file chunks. If a pirate shares the file chunks with another pirate, then it could potentially spread the poison. Therefore, it is critical to send poisoned chunks to pirates, not simply denying their requests. Otherwise, even if all clients deny pirate's requests, the pirate still can assemble a clean copy from those colluders who have responded with clean chunks. With poisoning, we exploit the limited poison detection capability of P2P networks and force a pirate to discard the clean chunks downloaded with the poisoned chunks. The rationale behind such poisoning is that if a pirate keeps downloading corrupted file, the pirates will eventually give up the attempt out of frustration.

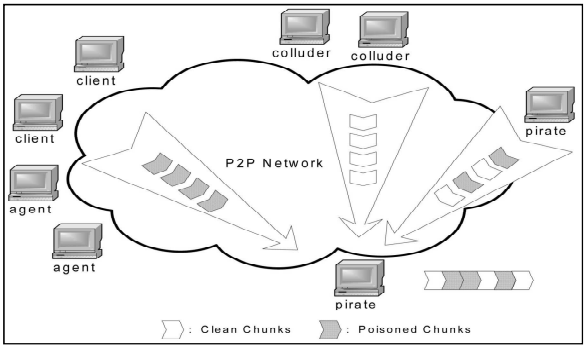


***Fig 6.*** *Proactive poisoning mechanisms built in trusted P2P network,where clean chunks (white) and poisoned chunks (shaded) are mixed infile streams received by pirates, but legitimate clients receive only clean Chunks.*

## 6. Conclusion and Future Enhancement

Here we conclude the Authority System to Prevent Privacy Protection in P2P Network system to stop paid peers and unpaid peers from suspected copyright male faction in file sharing network. And also when a pirate is detected, the distributed agent sends the falsified chunk file to the particular pirate client with proper counteractive actions. Combining DRM and reputation system to protect P2P content delivery networks will lead to a total solution of the online piracy problem. There are many other forms of online or offline piracy that are beyond the scope of this study. For example, our protection scheme does not work on a private or enclosed network formed by pirate hosts exclusively. We did not solve the randomized piracy problems using email attachments, FTP download directly between colluders, or replicated CDs or DVDs. In future we can focus on prototyping and benchmark experiments which are needed in Real-Life Open P2P Networks Here we can only prove the protection concept, lacking of sustained accuracy. Proactive chunk poisoning can be made selectively to reduce the processing overhead. However, further studies are needed to upgrade the performance of the copyright-protected system in real-life P2P benchmark applications.

## References

[1] N. Anderson(Sept. 2007), "Peer-to-Peer Poisoners: A Tour of Media- Defender," Ars Technica.

[2] S. Androutsellis-Theotokis and D. Spinellis (2004), "A Survey of Peerto- Peer Content Distribution Technologies," ACM Computing Surveys, vol. 36, pp. 335-371.

[3] S. Chen and X.D. Zhang(May 2006), "Design and Evaluation of a Scalable and Reliable P2P Assisted Proxy for On-Demand Streaming Media Delivery," IEEE Trans. Knowledge and Data Eng., vol. 18, no. 5, pp. 669-682.

[4] N. Christin, A.S. Weigend, and J. Chuang(2005), "Content Availability,Pollution and Poisoning in File-Sharing P2P Networks," Proc. ACMConf. e-Commerce, pp. 68-77.

[5] E. Damiani, D.C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante(2002), "A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks," Proc. ACM Conf. Computer and Comm. Security (CCS '02), pp. 207-216.

[6] M. Fetscherin and M. Schmid(2003), "Comparing the Usage of Digital Rights Management Systems in the Music, Film, and Print Industry,"Proc. Conf. e-Commerce.

[7] B. Gedik and L. Liu (June 2005), "A Scalable P2P Architecture for Distributed Information Monitoring Applications," IEEE Trans. Computers, vol. 56, no. 6, pp. 767-782.

[8] [T. Kalker, D.H.J. Epema, P.H. Hartel, R.L. Lagendijk (June 2004), and M. Van Steen, "Music2share—Copyright-Compliant Music Sharing in P2P Systems," Proc. IEEE, vol. 92, no. 6, pp. 961-970.

[9] B. Krishnamurthy, C. Wills, and Y. Zhang (Nov. 2001), "On the Use and Performance of Content Distribution Networks," Proc. Special Interest Group on Data Comm. on Internet Measurement Workshop (SIGCOMM).