International Journal of Computer Trends and Technology
ISSN: 2231-2803 / https://doi.org/10.14445/22312803/IJCTT-V68I1P110

Volume 68 Issue 1, 42-44, January 2020

Review Article

P2P Networks – Pirates Prevention

Sudha.V.S[1], Sharada M Kori[2], Gouri C K[3], Sharada G Kulkarni[4], Shubhda S Kulkarni[5], Pankaja B Patil[6]

[1,2,3,4,5,6,] *Assistant Professor, Department of CSE, Gogte Institute of Technology, Belagavi.*

Received Date: 20 November 2019
Revised Date: 15 January 2020
Accepted Date: 23 January 2020

Abstract *- P2P networks deal with delivering large files to a massive number of users. Hence piracy is the main source of violations of the content distributed within the boundary of a P2P network. The paid clients called the colluders illegally share copyrighted files with unpaid clients called pirates. We propose a technique to stop illegal file downloading by pirates. The pirates are detected by the use of time-stamped tokens assigned to each peer in the network. Detected pirates will receive poisoned chunks in their repeated attempts. We propose a content poisoning approach to stop copyright violation of the P2P network. Thus pirates are not provided with a chance to download the file successfully intolerable time.*

Keywords *- Content poisoning, P2P networks, Network security.*

I. INTRODUCTION

Collusive piracy is the main source of property violations within the boundary of a P2P network[1]. 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.

A P2P network is enhanced with the advantages of traditional content delivery networks where a large number of content servers was used over the globe. The content distributed replicated the contents on many servers.

P2P networks improve the availability of the content, and hence any peer can serve as a content provider. Because all peers are able to provide the content of the file, any illegal user can download the file from these peers by making a request.

Content poisoning[2] is implemented to respond to the pirate's request with poisoned chunks of the actual file requested by the pirate. Our scheme stops pirates from downloading files, even in the presence of colluding peers.

II. OUR APPROACH

We use a content poisoning technique to prevent collusive piracy that stops copyright violations in P2Pnetworks.

Each peer is identified with an endpoint address. A peer authentication protocol is developed that could be used by each peer to reveal itself as a legitimate peer. Thus an unauthorized peer is identified as a pirate as that peer fails to reveal itself as a legitimate peer.

Once a peer is identified as a pirate, its request will not be denied. Instead, all the clients will be sending poisoned chunks of the file. Finally, the pirate will be receiving a mixture of poisoned chunks from the paid clients and the clean chunks from the colluders. This will extend the download time of the pirate to a level beyond the practical limit.

III. P2P NETWORK ARCHITECTURE

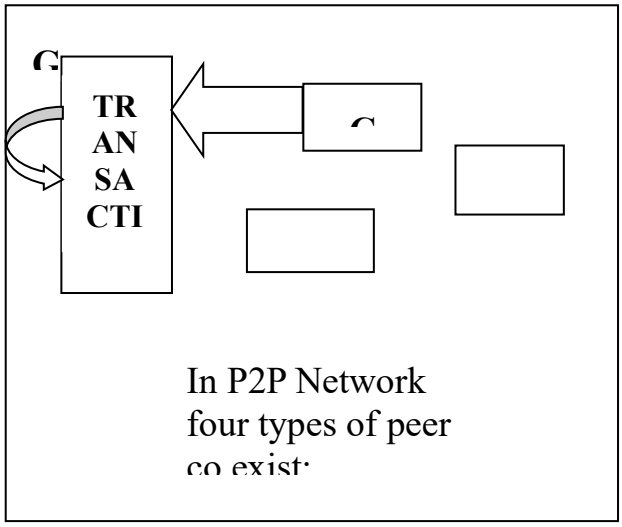
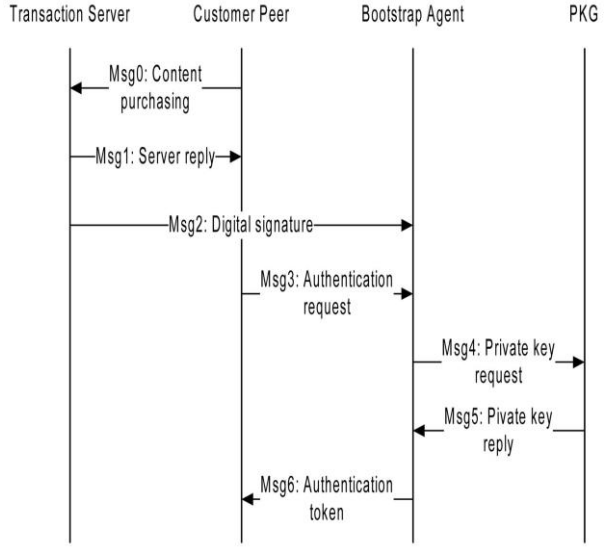


Fig. 1 A Trusted P2P Network

The network is built over a large number of peers. As illustrated in fig 1. there are four types of peers that coexist in the P2P network: clients (honest or legitimate peers), colluders (paid peers sharing contents with others without authorization), bootstrap agents (trusted peers operated by content owners for file distribution), and pirates (unpaid clients downloading content files illegally) To join the system, clients submit the requests to a transaction server that handles purchase content. The bootstrap agent generates a private key for secure communication among the peers. The transaction server and bootstrap agent are only used initially when peers are joining the P2P network. With IBS, the communication between peers does not require an explicit public key because the identity of each party is used as the public key.

Paid clients, colluders, and pirates are all mixed up without visible labels. Our copyright-protection network is designed to distinguish them automatically. Each client is assigned with a bootstrap agent as its entry point

A. Peer Joining Process



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 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:

Msg0: Content purchase request;
Msg1: Bootstrap Agent Address, E_k(digital_receipt, Bootstrap- Agent_session_key);
Msg2: Adding digital signature E_k(digital_receipt);
Msg3: Authentication request with userID, fileID, E_k (digital_receipt);
Msg4: Private key request with a private key request (observed peer address);
Msg5: PKG replies with privateKey;
Msg6: Assign the authentication token to the client.

Peers identify the pirates by checking the validity of extra signatures in file indexes. The trusted P2P applies this protection to share clean content exclusively among the peers and uses content poisoning techniques against the pirates.

IV. PEER AUTHORIZATION PROTOCOL

A. Token Generation

The PAP protocol consists of two integral parts: token generation and authorization verification. When a peer joins the P2P network, it first sends an 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, time-stamp} signed by the private key of the content owner. Since the bootstrap agent has a copy of the digital receipt sent by the transaction server, verifying the receipt is thus done locally. The Decrypt (Receipt) function decrypts the digital receipt to identify the file λ. The Observe (requestor) returns with the endpoint address p. The Owner Sign (λ,p, t_s) function returns with a token. Upon receiving a private key, the bootstrap agent digitally signs the file ID, endpoint address, and time-stamp to create the token. The reply message contains four tuples: {endpoint address, peer private key, time-stamp, token}. The reply message from a bootstrap agent is encrypted using the assigned session key.

Algorithm 1: Token Generation
Input: Digital Receipt
Output: Encrypted authorization token T
Procedures :
01: **if** Receipt is invalid,
02: deny the request;
03: **else**
04: **λ** = Decrypt(Receipt);
// λ is file identifier decrypted from receipt //
05: **p** = Observe(requestor);
// p is endpoint address as peer identity//
06: **k** = PrivateKeyRequest (p);
// Request a private key for user at p //
07: **Token T** = OwnerSign(f, p, t_s)
// Sign the token T to access file f //
08: **Reply** = { k, p, t_s, T}
// Reply with key, endpoint address, timestamp, and the token //
09: **SendtoRequestor**{Encrypt(**Reply**)}
// Encrypt reply with the session key //
10: **end if**

B. Peer Authorization Protocol

The PAP protocol is formally specified below. A client must verify the download privilege of a requesting peer before

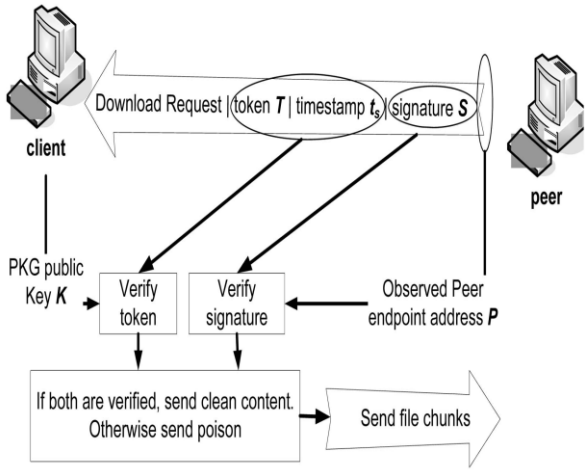


Fig. 3 The PAP detects a pirate upon illegal download request.

Clean file chunks are shared with the requestor. If the requestor fails to present proper credentials, the client must send poisoned chunks, as shown in Fig. 3. In PAP, a download request applies a token T, file index Ø, time-stamp

t_s, 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 t_s indicating the expiration time of the token. The Parse (input) extracts time-stamp t_s, token T, signature S, and index Ø from a download request. The function Match (T, t_s, 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 :
01: Parse (input) = { T, t_s, S, Ø(λ, p) }
// Check all credentials from a input request //
02: p = Observe(requestor);
// detect peer endpoint address p //
03: **if** {Match (S , p) fails},
//Fake endpoint address p detected //
return false;
04: **endif**
05: **if** {Match(T ,t_s ,K) fails},
return false;
// Invalid or expired token detected //
06: **endif**
07: **return true**;

V. CONCLUSION AND FUTURE WORK

Our protection scheme gives higher priority to satisfy honest clients. Putting poisoning tasks at lower priority reduces the upload overhead. With secure file indexing and assistance from a peer reputation system, colluding peers are detectable. The system stops piracy by poisoning pirates with excessively long download overhead in the presence of a large number of colluders. With the time-stamped authorization token, the PAP protocol enables clients to detect illegal download attempts from a pirate without communicating with a central authority. The proposed system is more effective to protect large files. The proposed PAP protocol detects colluders and pirates and applies chunk poisoning selectively. These extra activities add only limited extra workload or traffic to the network. These overheads are distributed among all distribution agents and clients, making their effects almost negligible on individual clients.

File-level reputation system posts a new challenge to work with DRM systems in P2P content delivery. The integration of selective poisoning with a reputation system and DRM will widen the CDN application domains. Combining DRM and reputation systems to protect P2P content delivery networks will lead to a total solution to the online piracy problem.

REFERENCES

[1] S. Androutsellis-Theotokis and D. Spinellis.,A Survey of Peer-to-Peer Content Distribution Technologies., ACM Computing Surveys
[2] N. Christin, A.S. Weigend, and J. Chuang.,Content Availability, Pollution and Poisoning in File-Sharing P2P Networks.
[3] RajkumarBuyya, Al-Mukaddim Khan Pathan, James Broberg and ZahirTari., A Case for peering of Content Delivery Networks
[4] B. Krishnamurthy, C. Wills, and Y. Zhang., On the Use and Performance of Content Distribution Networks.