# Pollution Attacks Identification in Structured P2P Overlay Networks

Zied Trifa[1(✉)], Jalel Eddine Hajlaoui[2], and Maher Khemakhem[3]

[1] MIRACL Laboratory, University of Sfax, Sfax, Tunisia
trifa.zied@gmail.com
[2] MARS Research Laboratory, University of Sousse, Sousse, Tunisia
hajlaouijalel.ig@gmail.com
[3] College of Computing and Information Technology, University of King Abdulaziz, Jeddah, Saudi Arabia
makhemakhem@kau.edu.sa

**Abstract.** Structured p2p overlay networks have emerged as a dominant means for sharing and exchange of information on the Internet. However, they suffer from severe security threats, known as pollution attacks, in which malicious peers insert decoys in data object. The existence of such polluters is considered as a major problem since these systems are based on trust between peers to ensure the sharing and access to available resources. Pollution attacks ravages network resources and annoys peers with contaminated objects. Although there have been numerous works on pollution attacks, there have been no studies on these attacks in structured p2p overlay networks and all of them are not qualified to ensure security. This paper investigates the different strategies of polluter nodes and their impact on the security of communication. We also detail a monitoring process to supervise, detect and attenuate these threats. Our experiments show that our strategy decreases enormously the pollution attacks with a slight number of monitor peers.

**Keywords:** Pollution attacks · Structured p2p overlay networks Chord · Monitoring · Tracking

## 1 Introduction

Structured p2p systems have grown increasingly in recent years as a means of communication, resource sharing, distributed computing and the development of collaborative application. They provide self-organization architecture of large-scale application. Thus, they were subjected to further analysis and a careful design to ensure scalability and efficiency [1].

However, recent research [2] have focused on creating efficient search algorithms that can be used to build more complex systems. But, they have not considered how to deal with pollution attacks. These attacks occur when a polluter peer added decoys in data object (Content pollution) or alters the metadata (Metadata pollution) or tries to falsify indexes (index poisoning). Thus, resulting in a wide range of polluted objects propagating in the system.

Pollution is one of the major issues affecting structured p2p networks. A study conducted in the KAD network to quantify the pollution of contents proved that 2/3 of the contents are polluted [15].

In this paper, our goal is to deal with pollution attacks in structured p2p systems using supervision and detection process. The remainder of this paper is organized as follows. In the next section, we provide some background information about pollution attacks. Section 3 reviews the related works. Section 4 details our contributions. We describe our proposed solution and its underlying ideas. In Sect. 5, we present details about the simulations steps and measurements used to assess the effectiveness of our supervision and detection process. Section 6 concludes the paper and outlines further directions.

## 2 The Pollution Attack

Pollution attacks damage targeted objects and dispatches them in the network. In this way, contaminated objects will be distributed through the sharing overlay. Thus, they break trust between users during objects exchange.

An object is considered polluted if the content does not fit the description presented to the user. Pollution attacks can be classified into three categories: Content pollution, Metadata pollution and Index poisoning.

The content pollution occurs when a malicious node adds decoys in data object. Thus, it can easily generate multiple false copies of objects that have the same content key by exploiting the weakness of the used hash functions. In this way, the transmission quality decreases significantly [3].

Metadata pollution occurs when polluter node alters the metadata of an object. Thus, nodes that will download objects based on metadata will obtain corrupted one [4]. In this way, nodes may unintentionally store contaminated objects in index table. It is very similar to the content pollution in terms of malicious intents. In both strategies, the polluter node tries to poison the content of the object to make it unusable. Thus, it uses its own resources to share contaminated objects in the overlay.

To find the location of desired objects, structured p2p systems use index. Polluter node tries to falsify these indexes by the insertion of massive numbers of false information. Consequently, when user attempts to download an object with randomly generated identifier, sharing system fails to locate the associated object.

Polluter node always tries to poison the index of the most popular objects. When other nodes download these objects, they get wrong or nonexistent one. Then, it connects directly to the victim's nodes. In this case, other nodes cannot obtain services from victim's nodes because these nodes have occupied the allowed connection [5].

Index poisoning directly attacks the structure of the overlay. First, polluter node can generate a random content key which could not point anywhere in the network. Moreover, it can generate multiple identities based on an invalid IP address or unavailable port number and publishes keys that point to one of it camouflaged identities [6].

## 3 Understanding Pollution Defense

Several researches have been done to address the pollution attacks in structured p2p overlay networks. In this section, we describe a wide range of mechanisms to attenuate these attacks.

### 3.1 Mechanisms Based on Downloading Objects

**Correspondence Techniques.** Correspondence techniques are based on the existence of a trusted centralized or decentralized database, which contains traces of the authentic objects. The authentic traces could be the content key or the metadata key. After downloading object, the node establishes correspondence with basic trust. If no match is found, node determines that the object is polluted. In this context, the project Sig2dat [7] makes available to users of KAZAA system a tool to obtain the authentic content key associated to any object in the network. This tool displays the titles of objects and key values on websites and forums.

**Filtering Techniques.** In these techniques, users must first check their downloaded objects before sharing their objects. In this way, the level of pollution attack would be significantly reduced. The major challenge is to provide users a robust system that encourages them to filter contaminated objects. Liang et al. [8] proposed an IP identification technique associated with malicious nodes. This is achieved by the use of special robots designed to collect metadata from the network.

### 3.2 Mechanisms Based on Trust, Reputation and Collaborative Approaches

**Reputation Techniques.** Kamvar et al. [9] have proposed EigenTrust: an algorithm that computes and maintains a reputation index for each peer in the network. This reputation is computed based on the experience of other peers, which interact with it. They have demonstrated how to use the index reputation to identify peers who provide contaminated objects. Costa et al. [10] have proposed Scrubber: a peer identifies malicious nodes that publish polluted objects on the basis of its experience and testimony of his neighbors. Vieira et al. [11] have proposed SimplyRep: a new decentralized reputation system that identifies and penalizes content polluters, while incurring in low overhead in terms of bandwidth consumption. It relies only on individual experiences of a peer to compute the reputation of its partners. Meng and Tan [12] have proposed a mechanism that computes pollution degree of each peer participating in the network. This degree is under the charge of its neighbors. When a query message is forwarded to the desired peer, its neighbors will calculate his pollution degree in order to assess that the file was polluted or not. Walsh and Sirer [13] have proposed Credence: a distributed reputation system, designed to thwart content pollution. It enables a peer to determine the authenticity of shared content. Peers in the Credence network votes on objects. The aim is to collate these votes and weight them by a novel similarity measure. Feng and Dai [14] have proposed Lip: a ranking approach based on the lifetime and the popularity of objects. They have proposed two detectors that filter logs files to identify contaminated objects. Zhang et al. [15] have proposed InfoRanking: a mechanism that tries to mitigate pollution attacks by ranking content items. It is based on the observation where malicious peers provide numerous fakes versions of the same information items in order to avoid blacklisting. Shin and Reeves [16] have proposed Winnowing: a novel distributed hash table based anti-pollution schema. It aims to reduce decoy index records held by DHT nodes in the system. Qi et al. [17] have proposed a reputation system combined with peer reputation and object reputation. They calculate the reputation of sharing objects by the reputation of the voting peer. Thus, honest peer, who uploads unpolluted objects and actively votes on objects, can have a higher reputation, while a malicious peer, who uploads polluted objects, would have a reduced reputation.

**Collaborative Techniques.** In these techniques, users download the objects from his neighbors who trust him completely. If a user starts receiving contaminated objects from any trust friend, it stops accepting objects and signals the presence of malicious users. These approaches allow users to locate their friends using instant presence detection process [18].

### 3.3 Mechanisms Based on Identification of Malicious Peers

The mechanisms of this category are based on the localization of malicious peers. Wang et al. [19] have proposed a schema based on messages in which the malicious peers can be rapidly located as long as they spread a single false message into the network. They try to track the origin of corrupted blocks. Gaeta and Grangetto [20] have proposed a monitoring tool to detect polluter nodes. They propose to use a statistical inference technique, namely Belief Propagation, to estimate the probability of peers being malicious. The detection algorithm runs by a set of trusted monitor nodes that receives notification messages from peers whenever they obtain a chunk of data. In [21] Gaeta et al. have proposed a system called DIP (*Distributed Identification of Polluters*) in p2p live streaming. DIP relies on checks that are computed by peers upon completing reception of all blocks composing a data chunk. A check is a message that contains the set of peer's identifiers providing blocks of the chunk as well as a bit to signal if the chunk is corrupted.

### 3.4 Discussion

Pollution attacks remain one of the major challenges to overcome especially in the context of structured p2p overlay networks. Unfortunately, reputation techniques were not effective in preventing or reducing such attacks. This is due to the complexity of setting such mechanisms in autonomous and complex systems. These are penalized if the peers realize bad votes. Besides, peer reputation mechanisms only care about the reputation of object providers, while object reputation mechanisms only care about the reputation of sharing objects. These mechanisms relay on identification-based approaches of malicious nodes. Herein, the major drawback is the high computational costs for verification all chunks and the communication overhead due to the number of messages exchanged between monitor nodes and nodes participating in the system.

We notice that all proposed solutions are not applied to structured p2p overlay networks and are not qualified to protect them in real time. We propose in the next section a new monitoring tool that can detect and isolate polluter nodes in real time. Our system is based on the identification of polluter nodes by monitoring the published messages.

## 4 Contributions

In this section, we present our vision of monitoring polluter nodes. It aims to detect a wide range of polluter nodes, which provide polluted objects.

### 4.1 Identifying Suspicious Polluter Nodes

The goal is to identify suspicious polluter nodes providing and pretending disposing polluted objects, yet narrow enough to exclude the vast majority of honest nodes. The main idea is to introduce monitoring peers within the overlay. Positioned in a strategic way, the monitors allow us to gain full control over a zone of the overlay. We use the Sybil attack to infiltrate the overlay and collect the different information of suspicious polluter nodes such as the IP address, the node identifier and port number. The aim here is to infiltrate the overlay with few number of monitor nodes, which are all controlled by one entity, the coordinator. These monitors seek to detect suspicious polluter nodes. The coordinator is able to create thousands of monitors on one single physical machine. We divided the overlay into zones to achieve accuracy and obtain a more global view. We introduce $2^n$ detectors into the network; the first n bits are different (prefix of each zone) and the following bits are fixed, they are the signatures of our detectors.

To infiltrate the network and detect suspicious polluter, the monitor M is implemented in the following steps. First, it sends hello message to the neighbor peers in order to poison their routing tables with entries that point our monitors. The peer that receive hello message will add the monitor to their routing table. Second, it sends lookup message to locate some random content IDs or random keyword IDs in the monitored zone. We must ensure that random content IDs or random keyword IDs does not exist in the ID space K. The normal behavior is to reply with the nearest nodes to the queried ID. However, the polluter puts its ID in the response and claims he is the owner of the queried ID. By checking who privileges the ownership of those non-existent IDs, we can identify suspicious polluter nodes. Finally, it gathers the following information: overlay ID, IP address and the port number of all suspicious nodes detected and report results to the coordinator. To bypass the detection process, the polluter node may behave appropriately or not respond to the search message. So we need to monitor the published messages to determine polluter nodes.

### 4.2 Monitoring Publish Message

**Infiltration Process.** We place a monitor peer within the suspicious polluter node spotted by the detection process for its best exploration. This enables us to control all the published and queried messages. At the start of the infiltration process, the monitor node introduces itself in the overlay in the following two steps. First, it initiates the monitor node and places next to the target node in the ID space:

$$\mathrm{ID_M} = \mathrm{Min}\,\delta\left(\mathrm{SP_i};\ \mathrm{M_j}\right) \tag{1}$$

Second, neighbor's discovery, a neighbor of a node M is any node that belongs to the transmission range of M. As soon as a monitor node M is infiltrated, it sends a hello message. Any node that receives the message and sends a reply back to M within a predefined time out will be added to its neighbor list.

**Monitoring Process.** For a node M to be able to monitor a node S. M must be a neighbor of both S and the neighbors of S, saying Ns. In such a case, M monitors all the communication of S and Ns. The monitor peer M captures information for each message sent and received from Ns to S in the following two steps. When suspicious peer S receives a request from the requester peer, it replies with monitor peer address because according to suspicious peer S routing table, monitor peer M is one of the closest peers to the requested ID. When the requester peer learns about the monitor peer, it sends the same request. Thus, the monitor receives a copy of all messages for the address space attributed to the suspicious peer S.

In the distributed hash table, the publication node publishes its sharing information using two types of messages: publish content message and publish keyword message.

In publish content message, requests are sent towards the hash of the object to associate an object with a source. In publish keyword message, requests are sent towards the hash of the keyword to associate the keyword with the object.

Monitor peer M should attempt to verify the content of each publish message and verify the content key in a keyword publish message. To achieve these goals, monitor peer determines first the nature of the publish message (content or keyword). If it is a publish content message, the monitor peer gathers the following information: the sender IP address and the port number, the object id, the content id, source IP address and the port number. Second, M verifies the location of the content id. If the IP address belongs to the blacklist nodes gathered by the identification process, M calls the isolation process. In the other case, if the IP address is valid, M invokes the verification of the published keyword message in order to verify the content message of each object id. Algorithm 1 details the pseudo code of the monitoring process.

If it is a publish keyword message, the monitor peer gathers the following information: the sender IP address and the port number, the keyword id and the list of object id. Also, for each object, M gathers the content id, IP address and port number. Second, M verifies the location of the keyword id. If the IP address belongs to the blacklist, the monitor calls the isolation process. However, if the IP address is valid, M verifies in the same way the location of each object id received to guarantee that IP address does not belong to the blacklist. Finally, M verifies the content id of each object through the verification of the packet information to determine if the object is polluted or not.

Algorithm 1. Monitoring publish message process

```
For each publish message (content, keyword)
    /*Publish Content message verification*/
    If (publish message = publish content)
        /* M stores (@IP, port number, object ID, content ID) */
        /* M Verify content ID location*/
        If (IP address of the destination ∈ list_suspicious_polluter_nodes)
            Call isolation process
        Else
            Call publish keyword message verification
        End
    Else
        /*Publish Keyword message verification*/
        If (publish message = publish keyword)
            /* M stores (@IP sender, port number, keyword ID, list of objects IDs) */
            For each object:
                /* M stores (@IP, port number, object ID, content ID) */
                /* M Verify keyword ID location*/
                If (IP address of the destination ∈ list_suspicious_polluter_nodes)
                    Call isolation process
                Else
                    /*Verify location of each object*/
                    For each object id
                        If (IP address of the destination ∈ list_suspicious_polluter_nodes)
                            Call isolation process
                        Else
                            /* M Verify content key, searches of content ID and verifies the packet information */
                            If (the packet information does not match with the packet information in the database)
                                Call isolation process
                            End
                        End
                    End
                End
            End
        End
    End
End
```

### 4.3 Isolation Process

Detection process is only the first step towards protecting the structured p2p overlay networks against polluter nodes. The notification and isolation process are used to propagate the notification of detected polluter nodes to the neighbors and takes the appropriate actions to isolate them from the overlay. To achieve these steps a monitor node executes the following actions. First, M sends to each neighbor of S an authenticated alert message in the following form:

$$\mathrm{ALT\_Msg} = \{\mathrm{ID_M};\ \mathrm{ID_{SP}};\ \mathrm{H_M};\ \mathrm{PK_M}\} \quad (2)$$

Second, each neighbor of S receiving the alert message achieves this three actions. It verifies the authentication of the alert message; marks S as a polluter node and stores the message in an alert buffer to prevent other nodes to accept or forward any message from and to S until its remove from the overlay. Finally, M proceed to the isolation process. It redirects all messages coming to S to other nodes; drops all messages forwarded by S and removes S from its neighbor list.

## 5 Evaluation

To evaluate our methodology, we performed several experiments on PeerfactSim.Kom [22] simulator. This tool has an advantageous architecture compared to other simulators and implements different distributed hash systems. Besides, we choose to implement our monitoring process using Chord protocol [23] since it's considered as the most deployed distributed hash table system. We reformed the application layer to incorporate polluter nodes. They intercept all search queries in order to claim the owner of the requested objects. Also, they spread polluted objects using the publication of content key or metadata key as described in Sect. 2.

In order to make statements on the performance of an overlay under pollution attacks, suitable scenarios are needed. We used two scenarios during the simulations. In the first, we used a network without any protection as depicted in Fig. 1. However, in the second, we activated our monitoring and detection process as shown in Fig. 2.

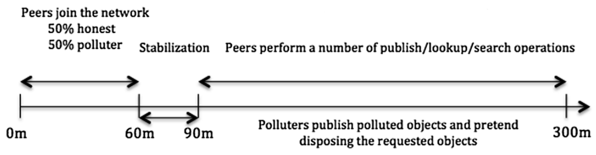
Peers join the network
50% honest
50% polluter
Stabilization
Peers perform a number of publish/lookup/search operations
0m
60m
90m
Polluters publish polluted objects and pretend disposing the requested objects
300m

**Fig. 1.** Pollution attacks without detection process

We considered a network with 500 nodes, 4 zones and 5 simulated hours. In the first step, each node joins the network. We assumed that 50% of nodes are honest and 50% are polluter. After stabilization phase, nodes perform random operations every

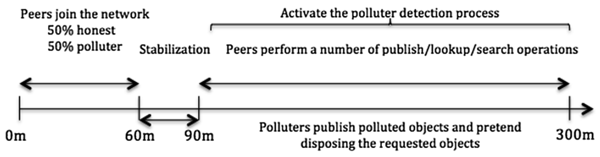
Peers join the network
50% honest
50% polluter
Stabilization
Activate the polluter detection process
Peers perform a number of publish/lookup/search operations
0m
60m
90m
Polluters publish polluted objects and pretend disposing the requested objects
300m

**Fig. 2.** Pollution attacks with detection process

60 s such as the publication and search objects. Honest nodes publish unpolluted objects. However, polluter ones publish polluted objects. Thus, they claim to be the source of all requested objects. Figures 3 and 4 present the evolution of the number of successfully and futilely published object.

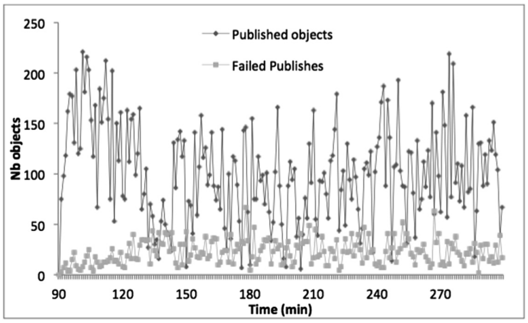

**Fig. 3.** The evolution of the number of successfully and futilely objects (without any protection)

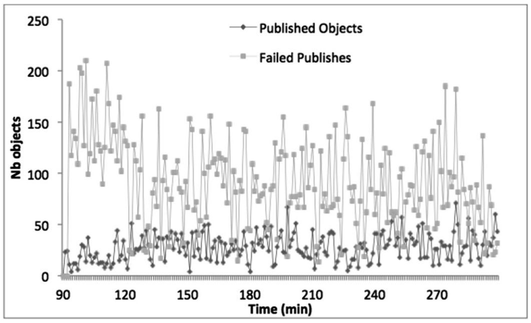

**Fig. 4.** The evolution of the number of successfully and futilely objects (with protection)

Figure 3 depicts the evolution of the number of successfully and futilely objects without any protection. However, we display in Fig. 4 the same number but after the activation of the protection process. In Fig. 3 we can notice that the number of successfully published objects is very important related with the number of failed one. Indeed, honest and polluter nodes publish objects in a random manner and the lack of a monitoring and control mechanism explain the high number of polluted and unpolluted object published successfully. Moreover, the malicious behavior of polluter nodes and the high complexity in the edifice of routing table explain the number of futilely objects.

In Fig. 4, we can notice that the number of successfully published objects decrease in a remarkable way. However, the number of futilely objects increases. This is due to the activation of the supervision and detection process. Finally, we note that the supervision has a lot of variations; this is due to the integration of the monitoring peers in the network and the variation of the malicious behavior peers. Besides, the dynamic nature of these peers causes a high change in the structure of the network.

Figure 5 presents the evolution of the number of monitored peers vs the evolution of the number of polluter peers when using a network with 4 zones. We can observe that the number of monitored peers raises exponentially, which due to the fact that the number of connected peers to our monitor peers increases over the duration of the experiment. Also, we can notice that the number of detected suspicious and polluter peers increases with the detection process. The high level of participating in the network, make polluter peers supervised and tracked by our tracking process.

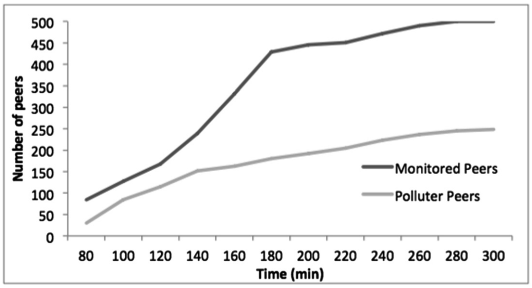

**Fig. 5.** The evolution of the number of monitored peers VS the evolution of the number of detected polluter peers

Finally, we present the evolution of the false negative and the false positive to assess the effectiveness of our monitoring process. Figure 6 depicts the evolution of the number of false negative related with the evolution of the number of suspicious peers. It refers to a failure to detect polluter peers that are present on a system. We can notice that the number of false negative decreases significantly in function of the evolution of suspicious peers.

Figure 7 shows the evolution of the number of false positive related with the evolution of the number of detected peers. It occurs when the detector peers mistakenly flag an honest peer as being infected. We can notice that the number of false positive is very low.

In summary, the validation experiments show that our supervision and tracking process detect close to 92% of polluter nodes, which prove the effectiveness of our methodology.

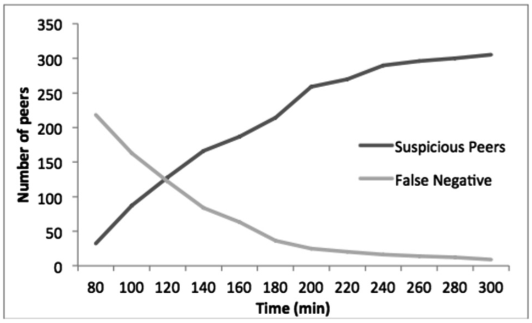

**Fig. 6.** False negative

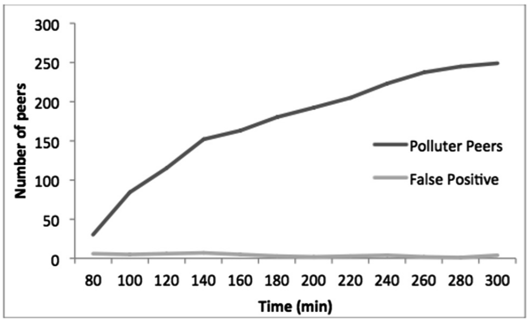

**Fig. 7.** False positive

## 6 Conclusion

In this paper, we presented the pollution attacks in structured p2p overlay networks. We have depicted that this attack is one of the major problems that affect these systems. Pollution attacks waste network resources and annoys users with contaminated objects. They damage the contents of the target objects and dispatches them in the network. In this way, contaminated objects will be distributed through the sharing system. We have proposed a new monitoring process based on three steps. The first step is based on the identification of suspicious nodes. The second step is based on supervision of all messages of suspicious nodes and its neighbors in order to identify polluter nodes and invoke the last step that allows the isolation process. Finally, we have implemented our methodology on the PeerfactSim.Kom simulator using the Chord protocol.

As a future work, we plan to implement our solution on some real distributed hash table such as KAD and try to refine both solution and the corresponding features in order to go further towards reaching a secure overlay networks.

## References

1. Maurya, R.K., Pandey, S., Kumar, V.: A survey of peer-to-peer networks. J. Adv. Res. Comput. Commun. Eng. (2016)
2. Liang, J., Kumar, R., Xi, Y., Ross, K.W.: Pollution in p2p file sharing systems. In: Proceeding of the International IEEE Conference INFOCOM, Miami, FL, March 2005
3. Chawla, S.: Content pollution in P2P system. J. Inf. Comput. Technol. **3**(8), 841–844 (2013)
4. Chen, C.S., et al.: Application of fault-tolerant mechanism to reduce pollution attacks in peer-to-peer networks. J. Distrib. Sensor Netw. **10**(7), 792407 (2014)
5. Locher, T., Mysicka, D., Schmid, S., Wattenhofer, R.: Poisoning the Kad network. In: Kant, K., Pemmaraju, S.V., Sivalingam, K.M., Wu, J. (eds.) ICDCN 2010. LNCS, vol. 5935, pp. 195–206. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11322-2_22
6. Liang, J., Naoumov, N., Ross, K.W.: The index poisoning attack in p2p file sharing systems. In: Proceeding of the International IEEE Conference INFOCOM, April 2006
7. Shi, J., Liang, J., You, J.: Measurements and understanding of the KaZaA P2P network. J. Current Trends High Perform. Comput. Appl. (2005)
8. Liang, J., Naoumov, N., Ross, K.W.: Efficient blacklisting and pollution-level estimation in P2P file-sharing systems. In: Cho, K., Jacquet, P. (eds.) AINTEC 2005. LNCS, vol. 3837, pp. 1–21. Springer, Heidelberg (2005). https://doi.org/10.1007/11599593_1
9. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The eigentrust algorithm for reputation management in p2p networks. In: Proceeding of the International Conference on WWW, Budapest, Hungary, pp. 640–651 (2003)
10. Costa, C., Soares, V., Almeida, J., Almeida, V.: Fighting pollution dissemination in peer-to-peer networks. In: Proceeding of the International Conference ACM SAC, Seoul, Korea, pp. 1586–1590 (2007)
11. Vieiera, A.B., et al.: SimplyRep: a simple and effective reputation system to fight pollution in P2P live streaming. J. Comput. Netw. **57**(4), 1019–1036 (2013)
12. Meng, X.-F., Tan, J.: Field theory based anti-pollution strategy in P2P networks. In: Yu, Y., Yu, Z., Zhao, J. (eds.) CSEEE 2011. CCIS, vol. 159, pp. 107–111. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22691-5_19
13. Walsh, K., Sirer, E.G.: Fighting peer-to-peer SPAM and decoys with object reputation. In: Proceedings of the International Conference on P2PECON, Philadelphia, August 2005
14. Feng, Q., Dai, Y.: Lip a lifetime and popularity based ranking approach to filter out fake files in p2p file sharing systems. In: Proceedings of the International Conference on IPTPS, February 2007
15. Zhang, P., Fotiou, N., Helvik, B.E., Marias, G.F., Ployzos, G.C.: Analysis of the effect of InfoRanking on content pollution in P2P systems. J. Secur. Commun. Netw. **7**(4), 700–713 (2014)
16. Shin, K., Reeves, D.S.: Winnowing: protecting P2P systems against pollution through cooperative index filtering. J. Netw. Comput. Appl. **31**(1), 72–84 (2012)
17. Qi, M., Guo, Y., Yan, H.: A reputation system with anti-pollution mechanism in P2P file sharing systems. J. Distrib. Sensor Netw. **5**, 44–48 (2009)
18. Montassier, G., Cholez, T., Doyen, G., Khatoun, R., Chrisment, I., et al.: Content pollution quantification in large P2P networks: a measurement study on KAD. In: Proceedings of 11th IEEE International Conference on Peer-to-Peer Computing, Japan, August 2011
19. Wang, Q., Vu, L., Nahrstedt, K., Khurana, H.: Identifying malicious nodes in network-coding-based peer-to-peer streaming networks. In: Proceedings of the International Conference on IEEE INFOCOM (2010)

20. Gaeta, R., Grangetto, M.: Identification of malicious nodes in peer-to-peer streaming: a belief propagation based technique. J. IEEE Trans. Parallel Distrib. Syst. **24**(10), 1994–2003 (2013)
21. Gaeta, R., Grangetto, M., Bovio, L.: DIP: Distributed Identification of Polluters in P2P live streaming. J. ACM Trans. Multimedia Comput. **10**(3), 24 (2014)
22. Graffi, K.: PeerfactSim.KOM – a peer-to-peer system simulator: experiences and lessons learned. In: Proceedings of IEEE International Conference on Peer-to-Peer Computing (2011)
23. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: a scalable peer-to-peer lookup service for Internet applications. In: Proccedings of ACM SIGCOMM, San Diego, California (2001)