2012-11-27

Fighting Pollution Attacks in P2P Streaming

Tauhiduzzaman, Md.

UNIVERSITY OF CALGARY

Fighting Pollution Attacks in P2P Streaming

by

Md. Tauhiduzzaman

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

November, 2012

Abstract

In recent years, the demand for multimedia streaming over the Internet is soaring. Due to the lack of a centralized point of administration, Peer-to-Peer (P2P) streaming systems are vulnerable to pollution attacks, in which video segments might be altered by any peer before being shared. Among existing proposals, reputation-based defense mechanisms are the most effective and practical solutions. In this thesis, we perform a measurement study on the effectiveness of this class of solutions. We simulate a framework that allows us to simulate different variations of the reputation rating systems, from the centralized global approach to the decentralized local approach, under different parameter settings and pollution models. In order to ensure that the framework and the simulated solution is representative enough, we dissect existing proposals and simulate a flexible defense mechanism, in which different components may be enabled and disabled by simply tuning certain parameters. Our experimental results reveal that global knowledge of the reputation rating is necessary to provide the best defense against the attack. But it is often susceptible under collaborative attacks, like collusion. We also find that expelling misbehaving peers is often more useful to prevent attacks than limiting their likelihood to be connected, although this can lead to poor playback quality. Based on these key observations, we propose DRank, a fully distributed rank-based reputation system, which decentralizes the global ranking system and combines it with Bayesian reputation rating systems. Experimental results show that this technique is more flexible and robust in fighting pollution attacks.

Acknowledgements

First, I would like to sincerely thank my thesis supervisor Dr. Mea Wang for her guidance and patience throughout this work. I thank my colleagues in the Network Research Group for helping me with resources and encouragements.

I am grateful to my parents for their sacrifices that have brought me up to this position. I thank my nearest friends Anik, Sabbir, Subashis vai, Mainur vai and Shubhrajit vai for making me feel at home here, thousands of miles away from my country, Bangladesh.

Finally, I thank Rifat for supporting me throughout this journey, believing in me no matter what, and being the light in the prolonged darkness.

Dedication

To my parents Md. Maniruzzaman Sarker and Nurun Nahar

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In recent years, the demand for multimedia streaming over the Internet is soaring. But the distribution of multimedia contents requires large amount of resources, which the small budget broadcasters cannot efford. For example, broadcasting videos in YouTube [1] costs Google millions of dollars per day to maintain the server space and upload bandwidth. To better accommodate the large demand and to increase the scalability of the service, the Peer-to-Peer (P2P) infrastructure is adapted. P2P systems utilize bandwidth from end-hosts and alleviate the workload on the content source. The idea of using P2P infrastructure to broadcast streaming media can ease the load on server, reduce cost and increase scalability, which are the main performance roadblocks of traditional client-server multicast systems. P2P streaming systems, *e.g.*, BBC iPlayer [2], PPLive [3] and UUSee [4], have been widely deployed to serve millions of users around the world. In a P2P live streaming system, at the source, the streaming file is divided into small *segments* representing a short duration in the video playback. The segments are then sent into the network and are shared among peers in the system. Due to the lack of a centralized administrative point, the segments might be altered by any peer before being shared. The system is under a *pollution attack* if unauthorized or unauthenticated information is inserted into the segments by one or more peers in the system.

Pollution attack in P2P file-sharing systems [5] is normally launched by copyright holders to fight copyright infringements. The same motivation works behind this attack in the P2P streaming systems, as experimented in [6]. Although this motimation makes this attack legal, the idea can be used illegally. Competing broadcasting companies can try to sabotage each other using pollution attack, or malicious users can launch this

attack just as a prank. This attack can even be launched unintentionally by poorly configured software at a machine in the network. Regardless of the motivation, pollution attack can be severe enough to make the entire system collapse with very little effort from the attacker.

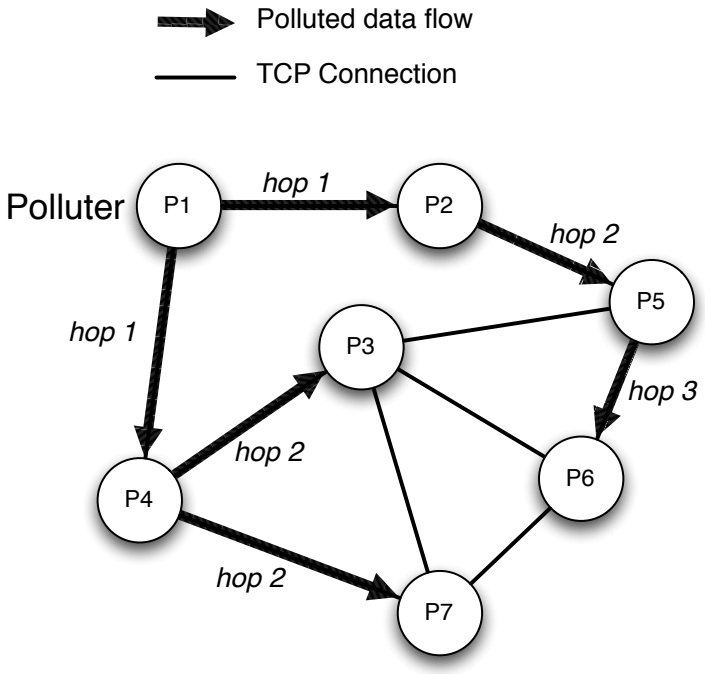


Figure 1.1: Polluted segment propagation

An example showing the propagation of a single polluted segment is illustrated in Fig. 1.1. The figure shows a small portion of a P2P streaming system. The polluter $P1$ produces a polluted segment and sends it to both $P2$ and $P4$. If these innocent peers are unaware of the pollution, they will relay the polluted segment to $P3$, $P5$ and $P7$. Thus, three more peers are being polluted. Finally, the segment reaches the only remaining peer $P6$ in 3 hops. The edges are labeled with the number of hops travelled by the polluted segment. This figure portrays the potential threats of the pollution attack. The polluter is able to contaminate all the peers in the example network without even being connected to all of them. In other words, anyone in the P2P streaming system can

launch such an attack. If there are multiple polluters with higher bandwidth available, they can easily pollute the entire system.

According to [5], a pollution attack is a simple but effective method to degrade the P2P system performance. In the KaZaA file sharing system, there may be 8000 to 50000 polluted versions of a single file in the entire network. According to [7], in file sharing systems like KaZaA, eDonkey and Gnutella, a small amount of polluted content can easily cause scarcity of the files. Polluters can also increase the severity of the attack by simply adding a few versions of a file to the network, even if they have very limited bandwidth [8]. A malicious peer, referred to as a *polluter* herein, can inject polluted content either aggressively or non-aggressively. A non-aggressive polluter downloads content as a regular peer and alters the content before sharing with others, whereas an aggressive polluter lures peers by simply advertising that it has all segments. While watching the video, viewers see either altered content or completely different video frames, leading to discontinuations of the regular playback or annoying video portions. As shown in [9] and [10], the entire network can be infected even by a single polluter in less than a minute.

The spreading of the polluted segments not only degrades the video quality, but also helps spread viruses, bots, and malware. Because of the anonymous and dynamic nature of the P2P infrastructure, it is difficult to identify the polluters. Polluters exploit the innocence of peers to distribute polluted segments widely. This also makes it difficult for the defense mechanisms to distinguish the polluters from victim peers. Expelling a victim peer will harm the streaming quality as the system not only loses bandwidth supply, but also good peers. In addition to that, the polluters may use intelligent approaches, *e.g.*, whitewashing and collusion attacks against any defense mechanism being applied. In a *whitewashing attack*, a polluter may frequently re-join the network as a new peer with a different IP address, and may also be less aggressive by sharing less bogus content in

order to keep its reputation within the acceptable range. In a *collusion attack*, a set of polluters collaboratively rank each other high and rank regular peers low in order to confuse the ranking system. These additional attack strategies make the pollution attack a burning issue.

To defend against such an attack, many solutions have been proposed, including cryptographic approaches, blacklisting, incentives, and reputation systems. The cryptographic approaches employ a hashing mechanism and distributed verification techniques to identify polluted contents as well as the source of the pollution [5, 9, 11]. They apply different encryption and signing techniques to protect transmitted segments from pollution attacks. These techniques include traffic encryption, hash verification, and chunk signing. Although the verification mechanisms can be applied in either a centralized way or a decentralized way, they require intensive computation either on the source node or among the peers in the network. In a P2P streaming session, video segments are being rendered by the video players as they are being received. It is a real challenge to deliver and verify the fingerprints and hash values in such a real-time system. Moreover, for this kind of defense, a peer must trust the source of the hash code [12], which can also be a malicious peer trying to provide wrong codes for bad segments. Monitoring trust in this system may require complicated defense structure.

In contrast to the cryptographic approaches, blacklisting [9, 13] is easy to implement and is very effective for straightforward pollution attacks. Peers that are exhibiting malicious behavior are blacklisted and avoided by regular peers. While this technique can effectively remove the polluters from the system, to counteract it, a polluter may combine a whitewashing attack or a collusion attack with the pollution attack. By far, the most effective and practical solutions for pollution attacks are the reputation-based rating systems [14, 15, 16, 17].

In reputation-based techniques, each peer is evaluated by the other peers with a rep-

utation value. In its simplest application, the peers store their observations locally as reputation values and choose segment sources based on these values. In global reputation-based techniques, the peer reputation values are stored globally. Each peer accesses these values to determine their sources. In the distributed techniques, the peers share their experiences with their neighbors in the form of local reputation values via gossiping to form a distributed structure of reputation storage. In this thesis, we first perform a simulation study on the effectiveness of the reputation-based defense mechanisms against pollution attacks. In contrast to existing measurement work [18], we compare different variations of the reputation systems, from a centralized rating system to a fully decentralized local rating system. We simulate the pollution model and the defense mechanisms using a simulator based on real TCP traffic. Our study is based on different pollution models that will not only pollute video segments, but also reputation ratings.

Based on our simulation, we observe the impact of key parameters, *e.g.*, threshold values and rating policies. Our study shows that each of the defense mechanisms have their advantages and limitations. For example, the global approach outperforms the local defense when the network is under a light pollution attack, and the local defense is more effective in coping with large systems and heavy attacks. In fact, we argue that a fully distributed defense mechanism is more suitable for the "real-time" P2P streaming system, since it is better in handling network dynamics and fast in detecting the polluters. To this end, we propose *DRank*, a fully distributed rank-based reputation system, which decentralizes the global ranking and combine it with Bayesian reputation rating systems. Experimental results show that this technique is more flexible and robust in fighting against pollution attacks.

We organize the rest of the thesis as follows. We compile the reviews of existing approaches against pollution attacks in P2P live streaming systems in Chapter 2. Chapter 3 provides a review of the P2P streaming systems and outlines the pollution models used

to test the reputation systems. In Chapter 4, we describe our simulated reputation systems, and evaluate their performance in Chapter 5. We propose DRank and assess its performance in Chapter 6. Finally, in Chapter 7, we conclude our work and discuss future works.

Chapter 2

Background and Related Work

According to [19], a reputation system performs a combined task of collection, distribution and aggregation of peer behavior observations to help decide on their honesty and trustworthiness. The reputation system has a wide range of applications in P2P and distributed systems, including trust management, system auditing, incentive building, and defence mechanisms.

2.1 Trust Management

Trust management is a burning issue in P2P systems. Users in popular retail services like eBay and Amazon demand trusted sellers to ensure quality products. For the same reason, P2P networks require efficient trust management to ensure that good content is shared. In P2P networks, data integrity can be ensured using packet signing [20], cryptographic fingerprints [21] or using message digests containing verification information provided by the server [22]. However, these techniques are bandwidth inefficient and/or susceptible to malicious peers that provide fake information to disseminate malicious content. Peers in P2P networks need to be trusted for proper implementation of the underlying protocol. It is also required to ensure the security of transaction data, and to protect resources like CPU and bandwidth from being used illegally [12]. Unlike online retail and social organizations, P2P networks do not have any centralized authority that can strictly maintain trust. Moreover, peers are anonymous and they act both as servers and clients, which makes them vulnerable to attacks like man-in-the-middle attack or spreading of viruses in the form of shared content [23].

Reputation-based trust management was originally introduced into P2P systems by Aberer *et al.* [24]. The primitive trust-management systems use simple binary reputation values (number of positive and negative transactions) as reputation ratings. These ratings are disseminated and aggregated using a simple polling approach [25, 26]. Before downloading content, peers poll their neighbors and decide on the source. These approaches, although straightforward and compatible with P2P networks, mostly rely on peer reviews rather than direct observations. This is susceptible to collusion and eclipse attacks. In more recent proposed trust management systems, instead of relying on simple polling from neighbors, the peers emphasize their own observations. A peer rates the behavior of its neighbors and shares the observations with other peers. The shared rates are incorporated into the own past experience of these neighbors to a certain fraction [27, 23, 28]. The trust information may stay within the neighborhood, and may also be further disseminated to the rest of the network. By aggregating the ratings, a peer can have a global view of the entire network [29]. The trust management system may exploit anonymity. Singh et al. [30] state that the anonymity of trust management in P2P networks is necessary to be consistent with the anonymity of P2P networks. They argue that anonymity of raters is important to protect peers from targeted attacks.

2.2 System Auditing

In system auditing, a reputation system relies on a single or a set of trusted peers who perform ratings on the rest of the system. The network formed by the trusted peers is commonly referred to as a *supervisory overlay* [31] and a *trust overlay* [32]. One common approach to the trust overlay is structured monitoring, in which peers are organized into a ring [33], like Chord, or multiple rings [34]. On a ring, the peers are organized to store reputation information to facilitate easy access by the peers [32]. This facilitates

the ease of information accumulation and storage with low cost. To relieve peers from extra tasks of reputation rating management, another type of auditing proposes that each peer is assigned a dedicated auditor. An auditor monitors the transactions of a peer and shares the information with other auditors. The effectiveness of the trust overlay in defending a P2P live streaming system is demonstrated by SecureStream [35], an auditor implementation based on Fireflies [34]. Fireflies is an efficient overlay structure capable of detecting live and dead peers when disseminating information. In addition to the local auditors assigned to each peer, there are global auditors that are in charge of evaluating reports from the local auditors. These auditors collect reports from the local auditors, determine whether the reports are authentic by comparing with other auditor reports, and decide on the malicious behavior of peers. Because of the globalization, the information dissemination can have fast impact on avoiding misbehavior. Alternatively, peers can be organized in a tree structure, as in [36], in which an auditor monitors the data flow into a random subtree. Another approach of auditing [37] proposes that the auditors randomly join the network as regular peers and monitor data flow by downloading and examining data from regular peers. Two separate trees are required for peers and auditors. Peer organization in tree structure is necessary to facilitate auditor join, and the auditor tree is required to make sure that the auditing does not affect streaming performance. A similar structured approach to prevent free-riding proposed in [38] enables peers to observe neighbor activities and report to a specific set of other peers using a certain interval. This approach is promising for P2P networks since it is fully distributed, and information dissemination enables gathering global information at each peer. It is also immune to collusion attacks since a peer aggregates information from a group of raters. Since a peer knows the group of information providers, it can easily detect a false reporter. Although such structured systems are very effective in identifying and isolating malicious peers, as shown in [39], it is not scalable and is vulnerable to network dynamics. It is

also susceptible to node failure and targeted attacks [40]. Application and maintenance of dedicated auditors and their overlays are costly. Furthermore, if the trust in the trust overlay is broken, the entire system is compromised.

2.3 Incentive Building

Reputation rating is also useful in building incentives for content sharing. Similar to the Tit-for-Tat sharing mechanism employed in BitTorrent systems, peers with good reputations have greater privilege in obtaining content from their neighbors. P2P networks highly rely on the cooperation of peers, which makes it susceptible to free-riders, who use the bandwidth of the network without sharing their own bandwidth by only downloading contents without uploading. While reporting reputation values, a peer may be untruthful because reporting good reputation values may downgrade its own reputation and raise competition [41]. Incentive approaches discourage this selfish behavior among peers [42, 43]. This approach can also be applied to reduce pollution dissemination, although it is not effective against certain type of pollution, *e.g.,* hash corruption [44]. The aforementioned auditing framework may be used here to govern the truthfulness of the rating [36].

2.4 Defense Mechanisms

Reputation systems are widely used for security purposes. The well-known real-life example is the customer rating system employed by eBay [45] and Amazon [46] to help customers choose better sellers. A simple binary feedback technique is used in eBay where a seller is evaluated by the summation of positive and negative feedback by the buyers. According to [47], the well-functioning of this technique is highly dependent on the raters' truthfulness. Aside from being untruthful quite often, raters rate sellers higher

or lower than they deserve, and sometimes they do not even rate the sellers. Because it is easy to change buyer-seller partnership, the raters can get away with it easily. Unfair ratings can be reduced using anonymity or cluster filtering [48]. Also, because this system is not highly dynamic and there is a central authority to control reputation, there is very little opportunity to launch intelligent and targeted attacks. The P2P networks, being highly dynamic in nature, cannot use this reputation system since it is susceptible to attacks like collusion [49], DoS [50], and eclipse [51]. These are common in P2P networks and defense against them requires a distributed solution frameworks like *Oversight* [50]. While concentrating on reputation systems, complex rating systems are used to identify DoS attacks [52], to detect eclipse and collusion attacks [33, 53], to fight SPAM and decoys [54, 55], and to defend the system against pollution attacks [14, 15, 16, 17]. Research shows that reputation-based defense can perform with any form of anonymity [56, 57]. Different reputation rating systems have been proposed to credit correct behavior and to penalize malicious behavior. The rating is used to determine the trustworthiness or to predict the cooperativeness of peers in the system.

In the reputation-based approaches to fight pollution attacks in P2P streaming systems, each peer evaluates its neighbors with a reputation value based on the quality of the segments received from them. These reputation values are then used to decide whether a peer is likely to provide unpolluted video segments. Storing and processing these reputation values are different in each proposal. In [14], these values reflect the correctness of the content flowing from neighbors to each peer. Each peer stores its own rating for all of its neighbors. No other information is used to rate a peer in the system, and different peers may have different ratings for the same peer. Although this solution is very scalable and is immune to collusion attacks, it takes time to determine the behavior of a peer, especially in a large and dynamic system. To this end, Buchegger *et al.* [17] proposes a Bayesian model that incorporates ratings of a peer shared by neighboring

peers into the calculation of the reputation values at each peer. In other words, the reputation value of peer j maintained at peer i is calculated based on both the first-hand information (content from peer j) and the second-hand information (rating about peer j at other neighbor peers). The second-hand information not only helps peer i to build a more general view of peer j, but also helps peer i to have a more accurate rating quickly, even without downloading from peer j.

Due to the powerfulness and effectiveness of the Bayesian model, we seek a deeper understanding of the model. Orthogonal to the work presented in [18] that identifies the proper use of a reputation system, in this thesis, we compare different variations of this model, from the centralized approach to the fully decentralized local approach, under different parameter settings and pollution models.

Chapter 3

Preliminary: Pollution Attacks in P2P Streaming Systems

The objective of this thesis is to study reputation-based defense mechanisms for fighting pollution attacks in P2P streaming systems. Before discussing reputation systems, this chapter provides an overview of a typical P2P streaming system, the pollution model and the simulator to simulate them.

3.1 P2P Streaming Systems

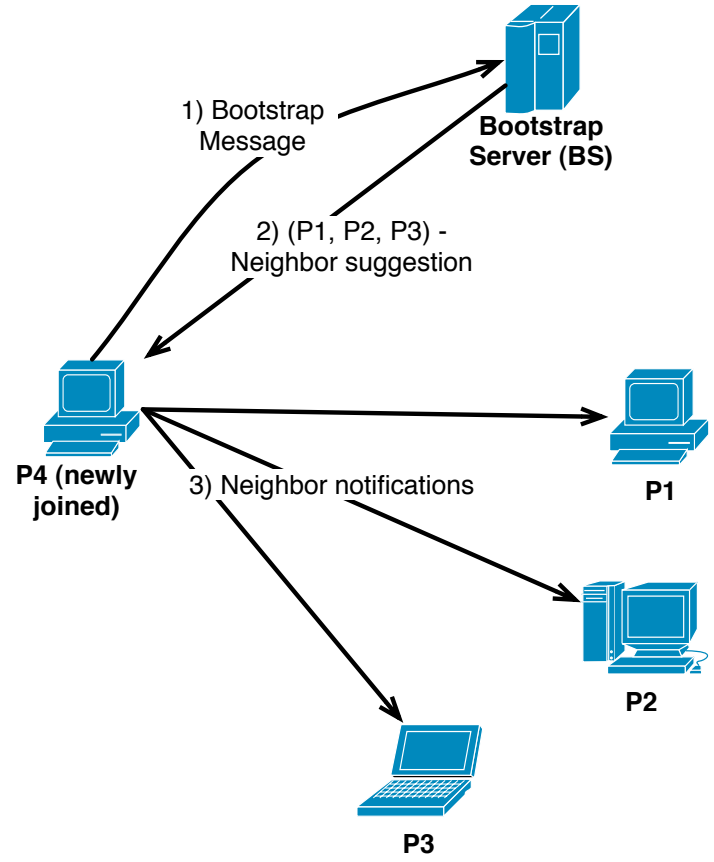
Existing P2P streaming systems can be categorized into two kinds: live streaming and Video-on-Demand (VoD). In a live streaming session, the content only becomes available as time evolves and is delivered to all peers at roughly the same time, much like TV broadcasting. In other words, peers share similar interests and form one large data swarm. In contrast, VoD is much like DVD playback. Peers may perform any DVD-like operations while viewing the video, including pause, fast forward, rewind, and scene selection. This means that peers may not necessarily share the same interests. In fact, peers whose interests overlap form a network to exchange segments. Essentially, a typical P2P VoD system consists of many small data swarms, separated by peers' interests. To better focus on the severity of pollution attacks and the effectiveness of the reputation system, we focus on the live streaming case since it has just one swarming session.

The successful deployments of PPLive [3] and UUsee [4] have demonstrated the practicality of pull-based live streaming systems, similar to the design presented in [58]. Since pollution attacks will only become a threat in a real system, we choose the pull-based streaming system as our target platform. This section provides a review of the pull-

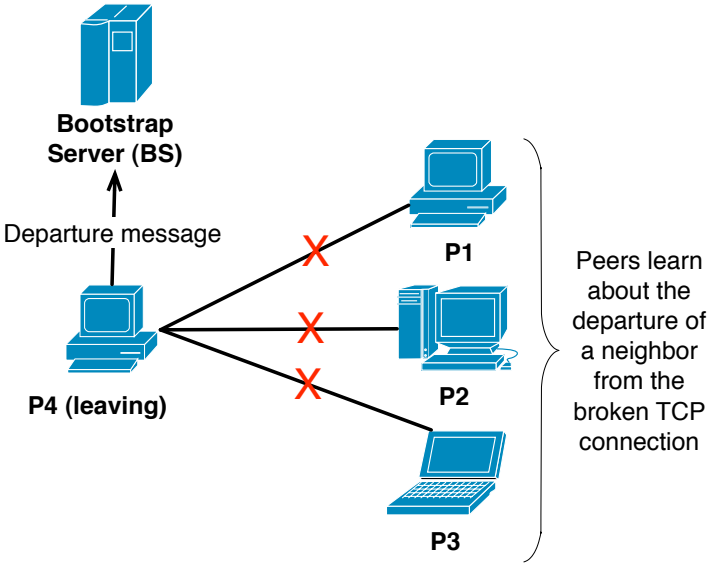
based streaming protocol, which is also the one we implemented in our simulator. The discussion will focus on the design choices for the bootstrap server, playback buffer, and segment scheduling.

As in many P2P systems, peers need a way to discover each other before forming a data swarm. Since the peer discovery algorithm is orthogonal to the defense against pollution attacks, we resort to the bootstrap server for maintaining peering relationships in the systems. The bootstrap server keeps track of the status of all peers in the system and suggests neighbors to each peer upon request. This server will later be extended to compute the global reputation rating. Upon joining the network, a peer contacts the bootstrap server for neighbor suggestions, and then tries to connect to the suggested peers, as shown in Fig. 3.1(a). The peers periodically send keep-alive messages to the bootstrap server so that the server will always have a global view of the present system. This allows the server to keep track of the number of connections served by each peer, which is the essential information for managing the load among peers. The server should ensure that the peers are not over-committing their bandwidth by maintaining too many connections. During a streaming session, each peer monitors the connection quality and the responsiveness of the neighbors. If a peer does not receive sufficient segments from a neighbor, it disconnects this neighbor and asks the bootstrap server for more suggestions. Peers in any P2P system may leave at any time. In our target system, we assume that when a peer leaves, it disconnects from all of its neighbors and notifies the bootstrap server, as shown in Fig. 3.1(b).

Fig. 3.2 depicts a small, but illustrative, example of a P2P streaming session. A *streaming server*, labeled S in Fig. 3.2, has the multimedia content to be distributed. It divides the multimedia content into *segments*, each representing a short duration of playback. As the segments become available, the source sends them to peers that are directly connected to it, which are peers $P1$ and $P2$. The rest of peers form a connected



(a) Peer bootstrapping at join



(b) Peer graceful departure

Figure 3.1: Peer join and leave facility

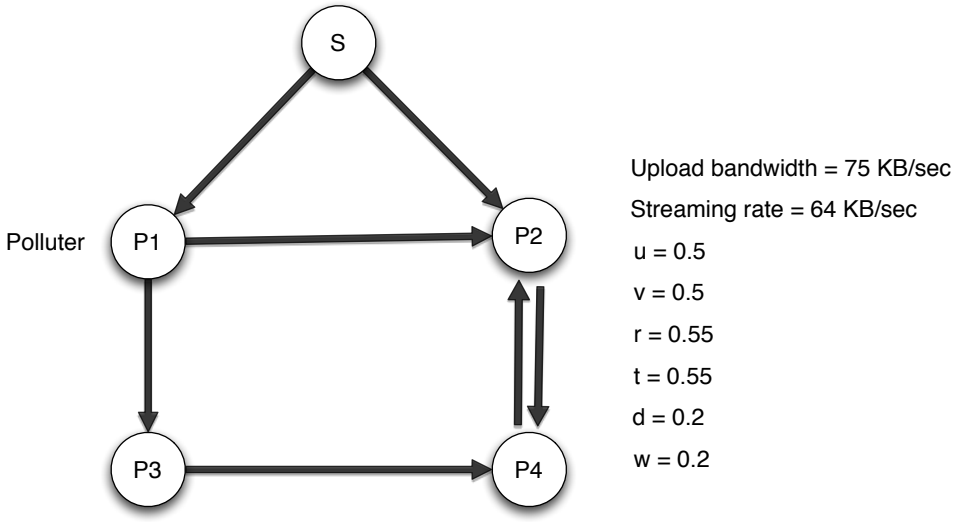


Figure 3.2: Static network for the running example of the local rating system

network based on the neighbor suggestions by the bootstrap server. They exchange segments with each other to form a P2P swarming session. A peer in our streaming system belongs to a certain neighborhood of peers. We classify the neighbors as upstream neighbors and downstream neighbors. A peer requests a segment from an upstream neighbor and serves a segment to a downstream neighbor. In the rest of the thesis, we address these two neighbors as *upstream peers* and *downstream peers* respectively.

Each peer maintains a *playback buffer* storing segments that are due for playback in the immediate future, as illustrated in Fig. 3.3. This buffer begins at the current playback point and ends at T seconds after the playback point. It marks the window of interest of each peer, *i.e.*, these are the segments that a peer is interested in receiving. For a live streaming session, the windows of interest among peers are roughly the same since the playback is closely synchronized. If a segment is received before its playback deadline, it is placed in the buffer at the appropriate slot. At the playback deadline, the segment is removed from the buffer to make room for future segments, and at the same time, the window of interests slides forward. If a segment is received after its playback deadline, or a copy has already been received, it is discarded by the peer, leading to a waste of

bandwidth.

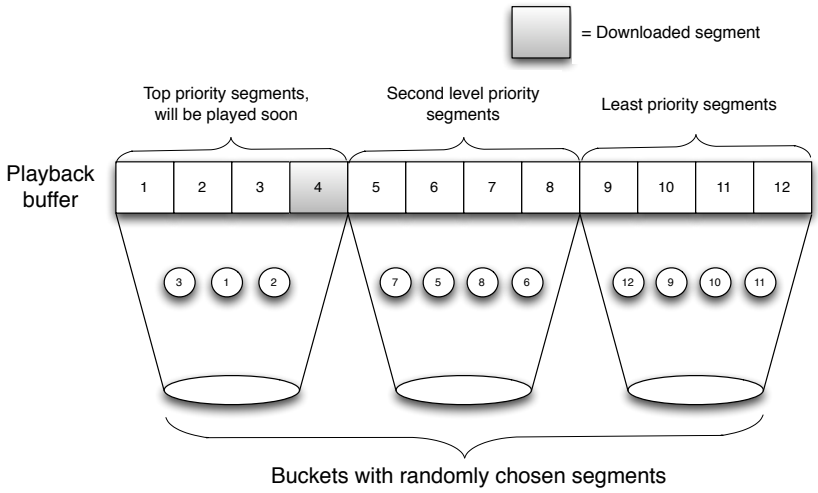


Figure 3.3: The playback buffer and the segment request scheduling

The primary task of a streaming protocol is to schedule segments for transmission with the goal to maintain smooth playback. Since segments must be played in sequential order, it is natural to expect that they arrive in this order too. However, in order to accommodate data swarming among peers, we also need to encourage diversity in segment availability in the peers' buffers. To do so, the streaming protocol for P2P streaming partitions the buffer into three prioritized regions, as marked in Fig. 3.3. The first region contains the slots for the segments whose deadlines are due very soon, and the last region consists of slots for segments at the end of the window of interest. Segments belonging to each region are put into a bucket for random selection, as illustrated in Fig. 3.3. When scheduling a segment for transmission, a peer starts from the first bucket, and will only move to the next bucket if the current one is empty. A peer randomly selects a segment from the non-empty bucket with the highest priority. This pseudo-sequential scheduling for segment transmission preserves the urgency of the segments with respect to their playback deadlines, while accommodating the data swarming to take advantage of the P2P infrastructure. Unlike file downloading, it is more important to provide peers with

a steady receiving rate and abundant bandwidth. For this reason, only one new segment is scheduled by a peer at each time unit represented by a segment to ensure a balance between the bandwidth supply and demand. A timeout mechanism is used to recover the unexpected long delays in transmitting the segments.

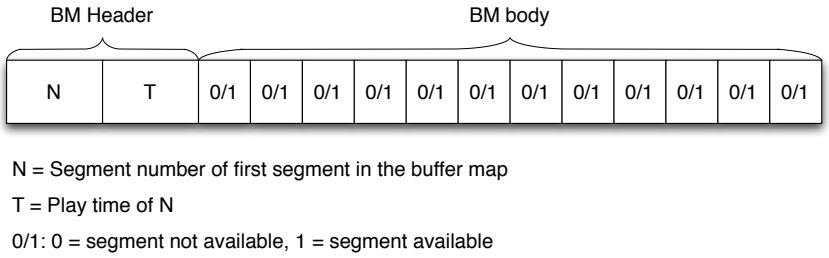


Figure 3.4: The structure of a buffer map

After neighborhood formation, the peers send buffer maps (BM) to their downstream peers. The BM contains a snapshot of the peers' playback buffer. This is the advertisement of the available segment at the peers. As shown in Fig. 3.4, the buffer map contains a header and a body. The header contains the segment number of the first segment in the buffer and its playback time. The inclusion of the play time is to ensure playback synchronization among the network. The body contains 1 bit for every slot in the buffer specifying the availability of segments at those slots. This bit is set if the corresponding segment is available. At the receiving side, the peer calculates the segment availability using the segment number at the header. If there are more than one sources of a segment, a peer randomly chooses a source to request. This ensures that the requests are distributed among all the sources of the segment. To avoid duplicated download of a segment, before re-requesting, each peer sends a segment cancel request to the previously requested upstream peer. To increase the number of segment sources, the peers aggressively try to connect to upstream peers by sending periodic suggestion requests to the bootstrap server, as long as the maximum upstream limit is not met.

The streaming session is initiated by the streaming server. It starts the streaming

by sending the first buffer map containing the first created segment information. The downstream peers of the server in turn send their buffer maps to their downstream peers. To assist this technique, we make sure that the peers and the streaming server send their buffer maps *after* each playback. We find from our experiments that with the above-mentioned streaming technique, the peers suffer less than 0.05% skips, which indicates smooth playback.

3.2 Pollution Models

In P2P streaming, a polluter may insert polluted content either *regularly* or *aggressively*. A polluter may act like a regular peer, meaning that it acquires video segments following the streaming protocol described in the previous section. Upon receiving each segment, the polluter modifies the content before making it available in its playback buffer for sharing. This attack is more challenging for the polluters since it must pull, modify, and send segments before their respective playback deadlines. Instead, a polluter can be more aggressive in launching the attack by acting like a source. Hence, it advertises a full buffer in its neighborhood and serves segments pulled by neighbors. In this case, the polluter does not need to connect to a server for the multimedia content. It either already has a modified copy of the media file or a completely different one. In this thesis, we resort to the aggressive model by default because it is more devastating, since it lures peers with false buffer map information. We will consider the regular model as a variation of the whitewashing attacks (defined later).

It terms of emulation, the polluter is a plug-in component of the streaming application. As illustrated in Fig. 3.5, the plug-in has a buffer that feeds polluted content into the playback buffer, from which segments are pulled by neighbors. When launching a non-aggressive pollution attack, the polluter participates in the data swarming as a regular

peer, but redirects all incoming segments into the *pollution buffer* first. It then decodes, modifies, and re-encodes each segment before making it available in the playback buffer. When launching the aggressive pollution attack, the polluter bypasses the data swarming and the segment modification process (the part drawn in dotted lines in Fig. 3.5). The pollution buffer is pre-populated with bogus data, which are then directly fed into the playback buffer.

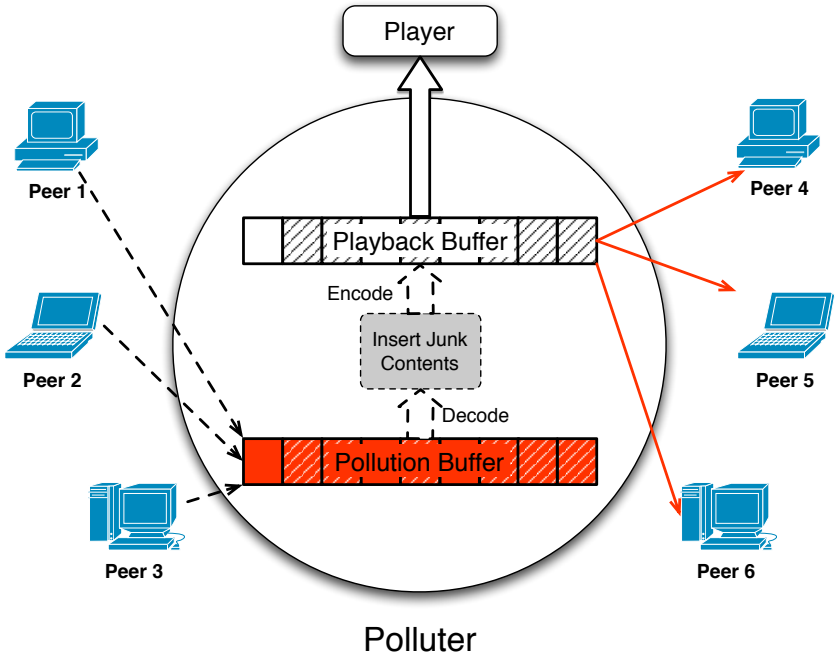


Figure 3.5: The process of pollution inside a polluter

We assume that a peer detects a downloaded polluted segment once it is played. At the player, when the segment is decoded, the peer can identify the junk contents. To assist this, we assume that a basic low-cost hashing is used. This hashing is different from the ones used to defend against pollution and it is integrated with the basic streaming. This means that, our simulated reputation-based model does not work alone. It may rely on other techniques to detect pollution to a certain extent.

Fig. 3.6 shows the local pollution detection process. At time t_0, the segment is buffered in the playback buffer and is scheduled to play at t_1. At time t_1, while attempting to

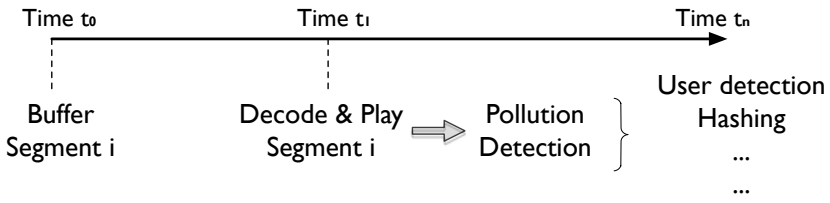


Figure 3.6: Timeline for pollution detection of a segment

play the segment, the peer uses either user observation or basic hashing techniques to detect pollution. We assume that a detection technique is already in place so that we can focus on the defense mechanisms. During the time between t_0 and t_1, the peer sends out this segment if it is requested from its downstream peers. So even if the peer can detect a polluted segment, it cannot discard it before disseminating. It forwards the segment being unaware of the pollution before the detection. For this reason, we need the reputation-based technique to detect polluters so that the pollution dissemination can be reduced.

Since the focus of this thesis is on the effectiveness of reputation-based systems in fighting pollution attacks, we should also consider the whitewashing and collusion effects, the two well-known techniques to pollute the reputation ratings in the system. In a *whitewashing attack*, a malicious peer tries to improve its image by changing its identity [59] or showing occasional good behavior (on-off attack) [60]. In this thesis, we assume that the polluter is aware of the reputation system and tries to improve its reputation ratings by sending a mix of good segments and polluted segments. By doing so, the polluter's reputation will not degrade much. Even worse, the direct neighbors of the polluters may have worse reputation than the polluters since it will not be able to participate in the data swarming with clean segments. This allows the polluters to hide in the system. This effect is similar to that of the non-aggressive attack described earlier. We simulate this whitewashing attack using a probabilistic model. Every time a polluter responds to a pull request, it sends a polluted segment with probability P. We tune the value of P in

our experiments to study the effectiveness of the reputation-based defense mechanisms.

Unlike the whitewashing attacks, the polluters collaboratively provide false reports on regular peers [49]. In this thesis, we simulate the collusion attacks in three forms: false positive, false negative, and both. In the *false positive* method, the polluters rate each other as well-behaving peers in the system and share that report with the rest of the system. In the *false negative* method, the polluters report all regular peers as polluters. We also combine the two methods so that the polluters will rate each other good and all other peers bad at the same time. Altogether, the aggressive model, whitewashing attacks, and collusion attacks provide complete coverage of pollution attacks, where the first one pollutes the content, and the latter two pollute the reputation system.

3.3 Emulator

The target of this thesis is to retrieve performance measurements as practical as possible. For this reason, we perform all our experiments in a custom-built emulator written in Java. The difference of this emulator with other existing simulators is that it enables data transmission with real TCP and UDP traffic. This provides the facility to perform close-to-reality experiemnts.

Data transmission with real traffic largely reduces the scalability of the emulator. So it is necessary to ensure efficient use of resources. In this emulator, each peer is presented with only two threads - one for basic socket programming and the other one for processing messages and the streaming application. We name these threads the *Network* thread and the *Application* thread respectively, as shown in Fig. 3.7. In a P2P network, each peer acts as both a server and a client. In a streaming system, a number of tasks need to be performed, *e.g.*, segment download, advertise, schedule and playback, in addition to simply form the network. These tasks require a number of network and application level

activities. To achieve realistic performance, the allocation of dedicated threads to each of these layers is necessary. Since the number of required threads at each peer is very small, this design is still scalable.

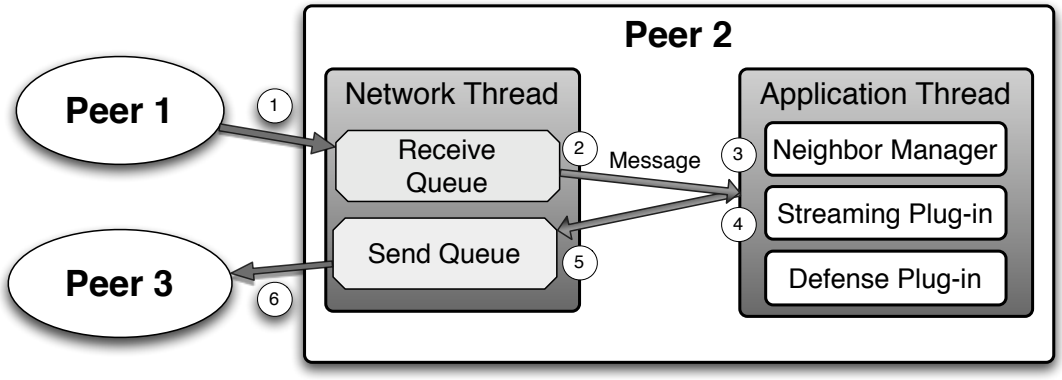


Figure 3.7: The emulator architecture and message processing: **Message receive:** 1) receive at network thread and enqueue 2) send to engine thread 3) process message. **Message send:** 4) generate and send to network thread 5) enqueue 6) send message

To facilitate the sending and receiving of messages, the *Network* thread in Fig. 3.7 maintains two separate queues - the sending queue and the receiving queue. These queues buffer the messages before sending and after receiving, respectively. The bandwidth limits are enforced on each connection to simulate heterogeneous communication links among the peers. The bandwidth limits are simulated as delays. Since the messages are sent on loopback connections or between machines in a high speed network, a silent period occurs after each successful send. The length of this period is exactly the time needed to transmit the message subject to the specific bandwidth limit, *i.e.*, it is the multiplication of the message size and the bandwidth. The *Application* thread in Fig. 3.7 is used to program applications required for network and streaming systems on top of basic data transmission facility.

In the *Application* thread, our simulated streaming system includes neighborhood management, streaming and defense techniques, as shown in the Fig. 3.7. In our simulated P2P streaming system, the neighborhood formation, segment dissemination and

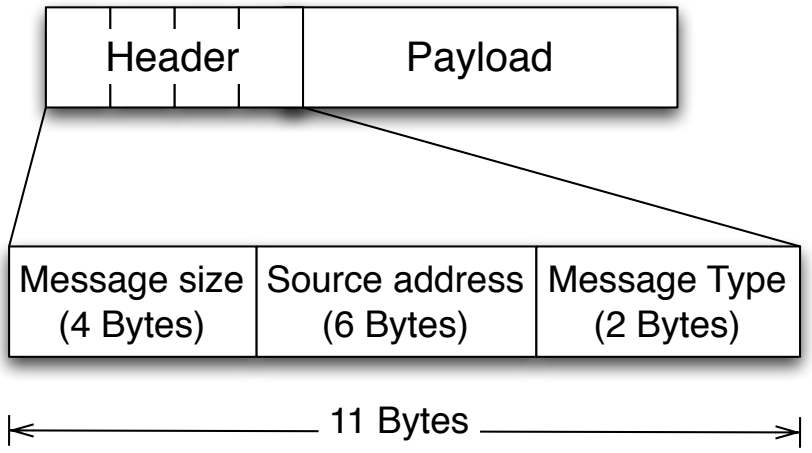


Figure 3.8: The message structure

playback are implemented here. These tasks are performed by sending task-specific messages. The messages are generated at the *Application* and sent to the *Network*. These messages include both control and data messages. The basic structure of each message is shown in Fig. 3.8. The header of the message contains the message size, source inforamtion and message type. Since the received message is processed only at the *Application*, the message size is required to facilitate delay calculation at the *Network*. The message type indicates what specific type of message it is. During the entire streaming session, a set of predefined message types are used. The size of the payload depends on the type of message. If it is a notification message, then the payload contains nothing. Other payloads contains data of different sizes. The largest messages are the data messages, since the entire segment is sent in a single message in our simulated system.

In our simulated P2P network, the bootstrap is initiated at the peers by sending out join request to the bootstrap server. For the streaming system, the streaming facility is simulated as a plug-in in the *Application* thread. The plug-in contains a playback buffer and a pollution buffer for the polluter implementation. The polluter is specified by simply setting a flag. The tasks of this plug-in are 1) generating and sending buffer maps, 2) scheduling segment requests, 3) playback and detect pollution and 4) for the

polluters, pollute segments. The defense techniques are also implemented as plug-ins. Each defense technique has its own set of methods coded in the plug-in. These methods are called at the streaming application for exploiting the defense to reduce pollution.

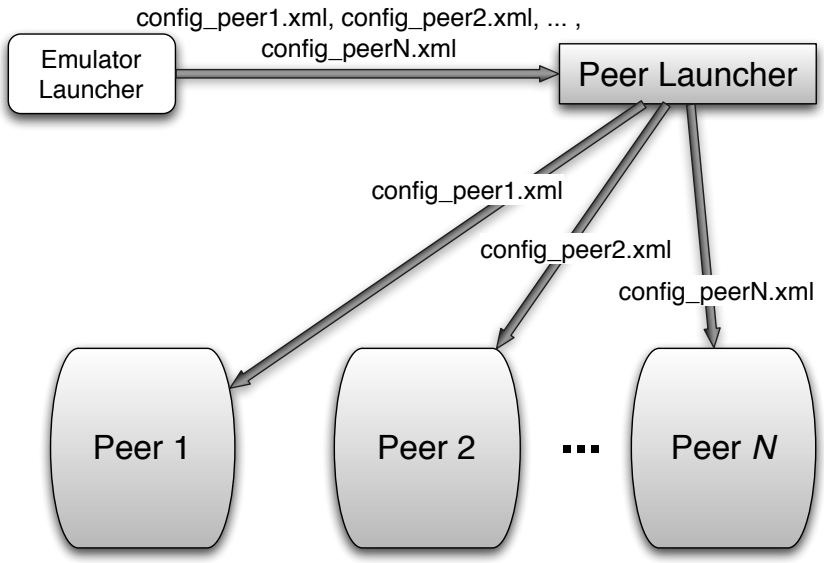


Figure 3.9: The distribution of peers among processes

Simulation of a large number of peers requires collaboration of multiple machines. Our emulator enables distribution of peers among multiple machines. Here, multiple processes are generated at each machine to create peers. Each machine is provided with a set of configuration files, using which the peers are generated. Each peer is provided with a configuration file containing the basic information about that peer, as shown in Fig. 3.9.

Chapter 4

Reputation-based Defenses in P2P Streaming Systems

Complementing the modeling work [39] and the analytical study [61], we seek to present a practical view of the effectiveness of reputation-based defense mechanisms. To do so, we must simulate different variations of the reputation-based defenses. The target of our study is to understand the effectiveness of different features of existing reputation-based techniques. For this reason, we extract the key components and simulate them in order to conduct a fair comparison. At one extreme, we present the ideal case where all first-hand information are available globally. An objective (normalized rating) view of all peers is made available by the monitoring station after pulling reports from peers. Here, we simplify the trust overlay model from [32] to simulate such a centralized approach to the rating system. At the other extreme, we study the fully decentralized case where each peer makes its own observation and decision independently. To this end, we examine the existing proposals [14, 15, 16, 17] and choose to implement the Bayesian model from [17]. The reason for choosing this model is three-fold. First, it shares a similar design with other works in obtaining first-hand information at each peer. Second, it is flexible to include second-hand information in peers' decision-making process. Third, it separates the reputation rating and the trust rating in order to infer the truthfulness of the second-hand information. In addition to simulating centralized and decentralized approaches, we analyze performance of the ranking and threshold approaches. The ranking approach is mainly used in the global approach and the threshold is used in both the global and the local approaches. In this section, we describe how we combine the Bayesian model and the trust overlay model to build a reputation system to fulfill our measurement goal.

Along with the mathematical model of the reputation system, we also provide a

numerical example based on the small example network presented in Fig. 3.2. We appoint $P1$ as the polluter. For each peer in a streaming session, it maintains reputation and trust rating for its direct upstream peers and downstream peers. For example, peer $P1$ in Fig. 3.2 rates the source S and two downstream peers ($P2$ and $P3$). As the second-hand information propagates through the network, peers will eventually have ratings for non-direct neighbors. The ratings will be summarized into a series of tables as we walk through the algorithm for updating the ratings, throughout the rest of this chapter. Each row in the table provides the reputation rating and the trust rating of peer P_j by peer P_i, with row header $P_i \rightarrow P_j$. A timestamp is associated with each table. However, the interval between the timestamps $(t_0, t_1, \ldots, t_n)$ may vary from one table to the next. Our goal is to illustrate each type of update with a numerical example. In the example network, we assume that each peer has upload bandwidth of $75KB/sec$, which is enough to support $64KB/sec$ streaming rate. Due to the small size of the example network, there is no need to assign the source extra bandwidth. The default values for the control parameters of the reputation system are given in Fig. 3.2. Their meaning will be defined throughout the next section.

4.1 Reputation and Trust Ratings at Peers

In the basic Bayesian inference, an entity gathers knowledge based on new events upon a prior knowledge about the system, thus facilitating efficient learning. This prior knowledge are commonly implemented with Beta Distribution. The entity updates the two parameters of the beta distribution based on new events. The amount of update is different in different systems based on the system protocol. We choose to use this Beta distribution for simulating the learning model for the defense techniques, as proposed in [17].

The Bayesian model in [17] was proposed to detect malicious nodes in P2P and mobile ad-hoc networks. The basic principle of the model is to enable the peers to obtain a global view of the network by exchanging information among themselves. For P2P live streaming, this information is the amount of unpolluted content served by each peer (the reputation rating) and the truthfulness of a peer (the trust rate), represented by two Beta distributions ($Beta(\alpha, \beta)$). To distinguish between the two ratings, we let $R_{i,j} = (\alpha_{i,j}, \beta_{i,j})$ and $T_{i,j} = (\gamma_{i,j}, \delta_{i,j})$ be the parameters for the reputation and the trust rating of peer j at peer i, respectively. The actual ratings are represented by the expected values of the distributions, which are denoted as $E(R)$ and $E(T)$ for reputation rating and trust rating, respectively. Now, we describe how these parameters are updated over time to reflect the pollution status in the network.

- When a peer joins the network, it forms a neighborhood with a portion of the peers in the network. Just after the neighborhood formation, the peer assigns $(1, 1)$ to both reputation and trust parameters. This is the prior value of the Bayesian system. For example, in our example network, all peers assign neutral ratings to their upstream peers and downstream peers, as depicted in Table 4.1.

- When peer i receives a segment from peer j, the reputation parameters for peer j is updated as follows:

$$\begin{aligned} \alpha_{i,j} &= u\alpha_{i,j} + s \\ \beta_{i,j} &= u\beta_{i,j} + (1 - s) \end{aligned} \tag{4.1}$$

  where s is 1 if the received segment is polluted and 0 otherwise. Here, we assume that a verification mechanism is in place to determine whether a segment is polluted. The constant $0 \leq u < 1$ is the discount factor that specifies how much flow history is kept in the decision making process. When u is close to 0, only the current observation is

Table 4.1: Reputation ratings at the peers at time t_0

	R: (α, β)	E(R)	T: (γ, δ)	E(T)
S → P1	1, 1	0.5	1, 1	0.5
S → P2	1, 1	0.5	1, 1	0.5
P1 → S	1, 1	0.5	1, 1	0.5
P1 → P2	1, 1	0.5	1, 1	0.5
P1 → P3	1, 1	0.5	1, 1	0.5
P2 → S	1, 1	0.5	1, 1	0.5
P2 → P1	1, 1	0.5	1, 1	0.5
P2 → P4	1, 1	0.5	1, 1	0.5
P3 → P1	1, 1	0.5	1, 1	0.5
P3 → P4	1, 1	0.5	1, 1	0.5
P4 → P2	1, 1	0.5	1, 1	0.5
P4 → P3	1, 1	0.5	1, 1	0.5

counted. When u is close to 1, the validity of each received segment will have a long term impact in the rating system.

Assume that during the time interval (t_0, t_1), the source S sent one good segment to each of $P1$ and $P2$, $P2$ sent one good segment to $P4$, and $P1$ sent one polluted segment to $P3$. The peers update their local reputation rating according to Eqn. 4.1. Each peer here keeps record of the sources of the segments so that when it detects pollution during playback, it can update the reputation ratings of the sources. The result is summarized in Table 4.2, in which '—' denotes that no changes to the table entry. We note that the actual reputation rating $E(R)$ decreases as more good segments are received, and increases after receiving each polluted segment. Even after just one round of segment transmission, the polluter already stands out with a high $E(R)$ value. We also note that good reputation rating corresponds to low $E(R)$, and bad reputation rating corresponding to high $E(R)$. To keep the discussion free of confusion, we will use the term *good/bad reputation* instead of *high/low reputation.*

- When no traffic flows from peer j to peer i for more than the predefined inactive period Δ, the reputation rating of peer j is discounted by a factor of u, *i.e.*,

Table 4.2: Reputation ratings at the peers at time t_1

	R: (α, β)	E(R)	T: (γ, δ)	E(T)
S → P1	—	—	—	—
S → P2	—	—	—	—
P1 → S	0.5, 1.5	0.25	—	—
P1 → P2	—	—	—	—
P1 → P3	—	—	—	—
P2 → S	0.5, 1.5	0.25	—	—
P2 → P1	—	—	—	—
P2 → P4	—	—	—	—
P3 → P1	1.5, 0.5	0.75	—	—
P3 → P4	—	—	—	—
P4 → P2	0.5, 1.5	0.25	—	—
P4 → P3	—	—	—	—

$$\alpha_{i,j} = u\alpha_{i,j}, \quad \beta_{i,j} = u\beta_{i,j} \tag{4.2}$$

Assume that from time t_0 to time $t_2 > t_1$, there was no traffic, except for the four edges carrying traffic during t_0 – t_1. If $t_2 - t_0$ is greater than the pre-defined inactive period, then the rating of the senders on these inactive edges will be updated by Eqn. 4.2. As shown in Table 4.3, the reputation rating $E(R)$ remains the same, but the values of α and β are greatly reduced.

- Each peer periodically broadcasts to its direct neighbors the reputation ratings for peers they observed since the previous broadcast. When peer i receives ratings of peer j from other neighbors, it incorporates the second-hand information into the rating of j as in Eqn. 4.3.

$$R_{i,j} = R_{i,j} + wF_{k,j} \tag{4.3}$$

where $F_{k,j}$ is the rating of peer j reported by neighbor k ($k \neq i$ and $k \neq j$), and $0 \leq w \leq 1$ is the weighting factor that defines how much second-hand information is incorporated

Table 4.3: Reputation ratings at the peers at time t_2

	R: (α, β)	E(R)	T: (γ, δ)	E(T)
S → P1	0.5, 0.5	0.5	—	—
S → P2	0.5, 0.5	0.5	—	—
P1 → S	—	—	—	—
P1 → P2	0.5, 0.5	0.5	—	—
P1 → P3	0.5, 0.5	0.5	—	—
P2 → S	—	—	—	—
P2 → P1	0.5, 0.5	0.5	—	—
P2 → P4	0.5, 0.5	0.5	—	—
P3 → P1	—	—	—	—
P3 → P4	0.5, 0.5	0.5	—	—
P4 → P2	—	—	—	—
P4 → P3	0.5, 0.5	0.5	—	—

Table 4.4: Reputation ratings at the peers after broadcast from S

	R: (α, β)	E(R)	T: (γ, δ)	E(T)
P1 → S	—	—	0.5, 1.5	0.25
P1 → P2	0.6, 0.6	0.5	—	—
P2 → P1	0.6, 0.6	0.5	—	—
P2 → S	—	—	0.5, 1.5	0.25

into the final rating of peer j.

Returning to our example network, assume that the source S broadcasts its local ratings about $P1$ and $P2$ to $P1$ and $P2$. Now, both $P1$ and $P2$ have second-hand information about each other. Since the local ratings and the second-hand information are the same, the update according to Eqn. 4.3 does not change the reputation rating $E(R)$ in Table 4.4.

The second-hand information is vulnerable to collusion attacks where polluters collectively rate each other very good and confuse the system with bad rating for regular peers. To defend against such an attack, a trust rating system is required to determine whether the reputation rating should include second-hand information. The trust rating of a peer

k ($T_{i,k} = (\gamma_{i,k}, \delta_{i,k})$) is updated in a similar way as the $R_{i,j}$ is. When a reputation report on peer j is received from neighbor k, the trust rating parameters for peer k at peer i are updated as follows:

$$\begin{aligned} \gamma_{i,k} &= v\gamma_{i,k} + s \\ \delta_{i,k} &= v\delta_{i,k} + (1 - s) \end{aligned} \tag{4.4}$$

where v is a constant similar to u in Eqn 4.1. The value of s is again binary, but this time is determined by a deviation test. The test first computes the difference between the current rating and the new second-hand information, and then checks the difference against a threshold value $0 \leq d \leq 1$, as in Eqn. 4.5.

$$|E(Beta(\alpha_{k,j}, \beta_{k,j})) - E(Beta(\alpha_{i,j}, \beta_{i,j}))| \geq d \tag{4.5}$$

where $E(Beta(*))$ is the expected value of a Beta distribution. The variable s is 1 if the report passes the test, and 0 otherwise. Following this update algorithm on the trust rating, the trust ratings of S on $P1$ and $P2$ are updated as in Table 4.4. Since the local reputation ratings and the second-hand information are identical on both peers, the trust rating for the source S is very good (low $E(T)$).

Let's examine another example where the polluter $P1$ broadcasts its ratings in the example network. The results are summarized in Table 4.5. When $P3$ receives a good rating about the source S from $P1$, it first runs the rating through the deviation test using Eqn. 4.5. Since $P3$ has no previous observation about the source S, its rating about S is neutral and the deviation test suggests to update the trust rating of $P1$. So the trust rating of $P1$ at $P3$ is updated to $E(1.5, 0.5) = 0.75$. Since this value is greater than the trust threshold 0.55, this value indicates the untrustworthiness of $P1$. As a result, $P3$ will not update its rating about S with this second-hand information. However, $P3$ will incorporate the rating about $P2$, $E(0.6, 0.6) = 0.5$, since $P3$ and $P1$ share the same rating $E(R) = 0.5$ for $P2$. Hence, the reputation rate of $P2$ is updated to $(1.12, 1.12)$, and the trust rating of $P1$ is updated to $E(0.75, 1.25) = 0.375$. Now if

Table 4.5: Reputation ratings at the peers after broadcast from $P1$

	R: (α, β)	E(R)	T: (γ, δ)	E(T)
S → P1	0.5, 0.5	0.5	0.25, 1.75	0.125
S → P2	0.62, 0.62	0.5	—	—
S → P3	1.1, 1.1	0.5	—	—
P2 → S	0.6, 1.8	0.25	0.5, 1.5	0.25
P2 → P1	0.6, 0.6	0.5	0.25, 1.75	0.125
P2 → P3	1.1, 1.1	0.5	—	—
P3 → S	1, 1	0.5	—	—
P3 → P1	1.5, 0.5	0.75	0.75, 1.25	0.375
P3 → P2	1.12, 1.12	0.5	—	—
P3 → P4	0.5, 0.5	0.5	—	—

$P1$, the polluter, tries to collude by broadcasting bad reputation values, *e.g.*, $E(99, 1) = 0.99$, about other peers, the polluter's trust rating will degrade since the reported value is very different from the locally stored value.

Furthermore, it is interesting to observe in Table 4.5 that there are several new entries that were not present in Table 4.1, *e.g.*, $S \rightarrow P3$ and $P2 \rightarrow P3$. As the broadcast information travels through the network, it also allows peers to discover other peers and learn their reputations without becoming direct neighbors.

Similar to the reputation ratings, the trust ratings of a peer are discounted by a factor of v if no new report is received for the duration of each inactive period. The reputation and trustworthiness of a peer are the expected values of the respective Beta distributions, *i.e.*,

$$\begin{cases} E(Beta(\alpha, \beta)) < r & normal \\ E(Beta(\alpha, \beta)) \geq r & misbehaving \end{cases} \tag{4.6}$$

$$\begin{cases} E(Beta(\gamma, \delta)) < t & trustworthy \\ E(Beta(\gamma, \delta)) \geq t & untrustworthy \end{cases} \tag{4.7}$$

where r and t are the thresholds for each rating. The second-hand information is only incorporated, *i.e.*, applying Eqn. 4.3, if the reporting peer is trustworthy.

Now, assume that the first round of reputation rating calculation ends at time t_3. The

Table 4.6: Reputation ratings at the peers at time t_3

	R: (α, β)	E(R)	T: (γ, δ)	E(T)
S → P1	0.62, 0.62	0.5	0.25, 1.75	0.125
S → P2	0.62, 0.62	0.5	0.125, 1.875	0.0625
S → P3	1.22, 1.22	0.5	1, 1	0.5
S → P4	1.1, 1.1	0.5	1, 1	0.5
P1 → S	0.62, 1.82	0.254	0.5, 1.5	0.25
P1 → P2	0.824, 0.824	0.5	0.125, 1.875	0.0625
P1 → P3	0.62, 0.62	0.5	0.625, 1.375	0.3125
P1 → P4	1.225, 1.225	0.5	1, 1	0.5
P2 → S	0.6, 1.8	0.25	0.5, 1.5	0.25
P2 → P1	0.848, 0.848	0.5	0.25, 1.75	0.125
P2 → P3	1.2, 1.2	0.5	1, 1	0.5
P2 → P4	0.5, 0.5	0.5	0.625, 1.375	0.3125
P3 → S	1.3, 1.3	0.5	1, 1	0.5
P3 → P1	1.5, 0.5	0.75	0.75, 1.25	0.375
P3 → P2	1.12, 1.12	0.5	1, 1	0.5
P3 → P4	0.62, 0.62	0.5	1.375, 0.625	0.6875
P4 → S	1.5, 1.5	0.5	1, 1	0.5
P4 → P1	1.244, 1.244	0.5	1, 1	0.5
P4 → P2	0.5, 1.5	0.25	0.375, 1.625	0.1875
P4 → P3	0.5, 0.5	0.5	1.125, 0.875	0.5625

reputation ratings at all peers are presented in Table 4.6. We note that, because of the small values of α and β, peer $P3$ detects the polluter using Eqn 4.6 just after receiving one polluted segment. However, as time passes, the values of α and β become larger, which makes this detection process longer. In summary, we have seen examples that this defense mechanism can cope with the collusion attacks. Since the reputation values are updated after receiving a segment, and broadcasting ratings quickly spread information in the system, this mechanism should also be effective when polluters are performing whitewashing attacks. We will evaluate the effectiveness of local defenses in detecting the polluter in Chapter 5.

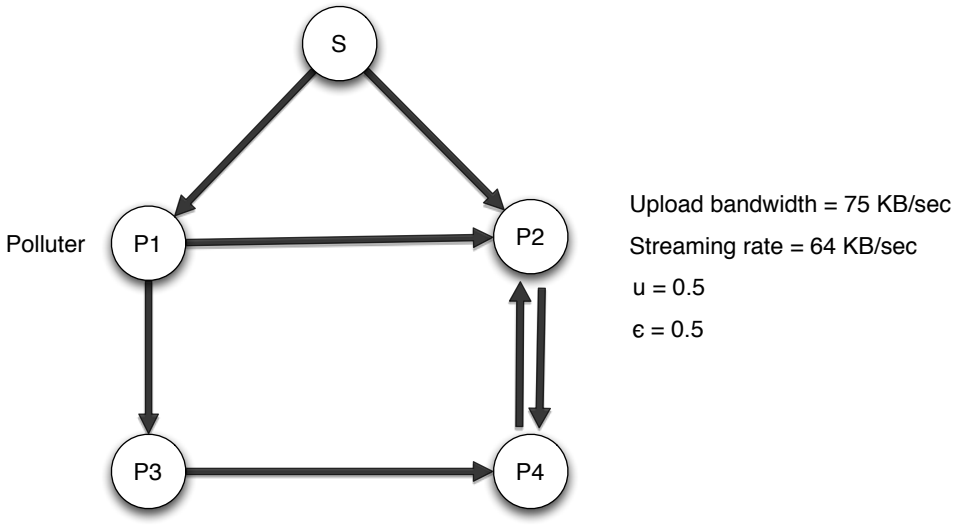


Figure 4.1: Static network for the running example of global approach

Table 4.7: Reputation ratings at the peers at time t_0

	R: (α, β)	E(R)
S $\rightarrow$ P1	1, 1	0.5
S $\rightarrow$ P2	1, 1	0.5
P1 $\rightarrow$ S	1, 1	0.5
P1 $\rightarrow$ P2	1, 1	0.5
P1 $\rightarrow$ P3	1, 1	0.5
P2 $\rightarrow$ S	1, 1	0.5
P2 $\rightarrow$ P1	1, 1	0.5
P2 $\rightarrow$ P4	1, 1	0.5
P3 $\rightarrow$ P1	1, 1	0.5
P3 $\rightarrow$ P4	1, 1	0.5
P4 $\rightarrow$ P2	1, 1	0.5
P4 $\rightarrow$ P3	1, 1	0.5

4.2 Global Reputation Ratings

In [32], a defense mechanism was proposed to improve content availability in the P2P file sharing applications. It is a distributed approach to global ratings to provide a uniform and global view of peer reputation. A centralized system, achieved by the collaborative effort among a set of trusted nodes, collects observations reported by peers. The global knowledge enables the computation of the peer reputations to reflect the true reputation of peers in the network, with respect to others. In contrast to the local ratings maintained at each peer, the global ratings are not just a combination of the observations of a neighborhood, it is a consensus of the entire network. Thus, decisions are more accurate and effective. Peers in the network tend to connect to peers with good global reputation. Thus, misbehaving peers are avoided as much as possible.

In our implementation, peers still independently compute reputations of their neighbors as described in Sec. 4.1. The peers periodically send these ratings to a server that represents the trust overlay. We simplify the trust overlay to a single server to better focus on the effectiveness of the reputation system itself rather than various implementations. Assuming that there are N peers in the network, the server maintains an $N \times N$ matrix (M), in which each entry $M_{i,j}$ is the reputation rating of peer i reported by peer j. We refer to this matrix as the *reputation matrix*. After receiving a new report from a peer, all corresponding entries in M are updated with the new data.

To illustrate this concept through an example, we update our example network, as in Fig. 4.1, with a new set of control parameters that are specific to this defense.

Before the streaming begins, at time t_0, the reputation ratings at each peer are initialized to the neutral rating, as in Table 4.7. This reputation approach does not involve any trust rating and broadcast mechanism, since all information is centrally maintained at the server.

Assume that during the time interval (t_0, t_1), the source S sent one good segment to each of $P1$ and $P2$, $P2$ sent one good segment to $P4$, and $P1$ sent one bad segment to $P3$. The updates in the local rating among peers are presented in Table 4.8. This table is the same as the reputation ratings from Table 4.2, since the algorithm for local reputation rating remains

Table 4.8: Reputation ratings at the peers at time t_1

	R: (α, β)	E(R)
S → P1	—	—
S → P2	—	—
P1 → S	0.5, 1.5	0.25
P1 → P2	—	—
P1 → P3	—	—
P2 → S	0.5, 1.5	0.25
P2 → P1	—	—
P2 → P4	—	—
P3 → P1	1.5, 0.5	0.75
P3 → P4	—	—
P4 → P2	0.5, 1.5	0.25
P4 → P3	—	—

the same.

After receiving reports from all peers in the network, the server updates its reputation matrix to the following:

$$M = \left(\begin{array}{c|ccccc} & S & P1 & P2 & P3 & P4 \\ \hline S & -1 & 0.25 & 0.25 & 0.5 & 0.5 \\ P1 & 0.5 & -1 & 0.5 & 0.75 & 0.5 \\ P2 & 0.5 & 0.5 & -1 & 0.5 & 0.25 \\ P3 & 0.5 & 0.5 & 0.5 & -1 & 0.5 \\ P4 & 0.5 & 0.5 & 0.5 & 0.5 & -1 \end{array}\right)$$

Since the peers do not report about themselves, the diagonal of the matrix are filled with -1's. Based on the reputation matrix M, the server periodically computes the global reputation values of all peers as follows:

$$G_i = \epsilon * G_i + (1 - \epsilon) * Avg_i \tag{4.8}$$

where G_i is the weighted average of the current reputation rating of peer i and the

average rating of peer i collected from peers (Avg_i), and $0 \leq \epsilon \leq 1$ is the weighting factor. The Avg_i is calculated as the weighted average of all reputation ratings of peer i, where the weighting factor is the reputation of the respective reporting peers, as in Eqn. 4.9.

$$Avg_i = \sum_{j=1, j \neq i}^{N} G_j M_{i,j} \tag{4.9}$$

The G_i's are maintained in a vector and are normalized so that $\sum G_i = 1$. Initially, before any report is sent to the server, the global rating of each peer is set to $\frac{1}{N}$. For example, in the example network, at time t_0, the global ratings are as follows:

$$G = \left(\begin{array}{ccccc} S & P1 & P2 & P3 & P4 \\ \hline 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \end{array} \right)$$

At time t_4, after receiving global ratings from all peers, the calculated normalized global ratings are:

$$G = \left(\begin{array}{ccccc} S & P1 & P2 & P3 & P4 \\ \hline 0.17 & 0.22 & 0.19 & 0.21 & 0.21 \end{array} \right)$$

The peers are then ranked according to G_i. The smaller the G_i is, the higher peer i is ranked. Rank 1 is the lowest rank, and rank N is the highest ranking, *i.e.*, the most promising peer in the system. When a peer contacts the server for a list of neighbors, the server searches the ranked list from the top or filter the list according to a threshold value for availability and suggests peers accordingly.

Chapter 5

Performance Analysis

In order to gain a deeper understanding of the reputation-based defense mechanisms for pollution attacks in P2P streaming systems, we simulate a streaming system, several pollution models, and the basic components described in Chapter 4. Furthermore, we believe that the most insightful results are achieved with a close-to-reality experiment. Therefore, we simulate all network communication and data transmission with real TCP traffic. However, this greatly reduces the scalability of the simulator. With our available resources, we can simulate only 200 peers. Since the focus of this work is on the effectiveness of the reputation system and the impacts of the key parameters on the streaming quality, we argue that it is more important to conduct the experiments with real traffic in real time than in a large synthetic network. In this chapter, we first describe the simulation settings and then define the performance metrics of our interest. We begin the analysis with parameter tuning for both the global reputation rating system and the local reputation rating system. We then compare the effectiveness of the two rating systems under different settings and pollution models.

5.1 Simulation Setup

The standard network consists of a server (the streaming source, the reputation server, and the bootstrap server) and 199 peers, out of which some are arbitrarily selected to act as polluters. By default, we select 10% of the peers to be polluters. Each polluter pollutes the network in an aggressive way, *i.e.,* advertising the availability of all segments without the need to retrieve any content from the source. As stated in Chapter 3, we

consider the live streaming case since it allows us to focus on the defense mechanisms against more challenging pollution models.

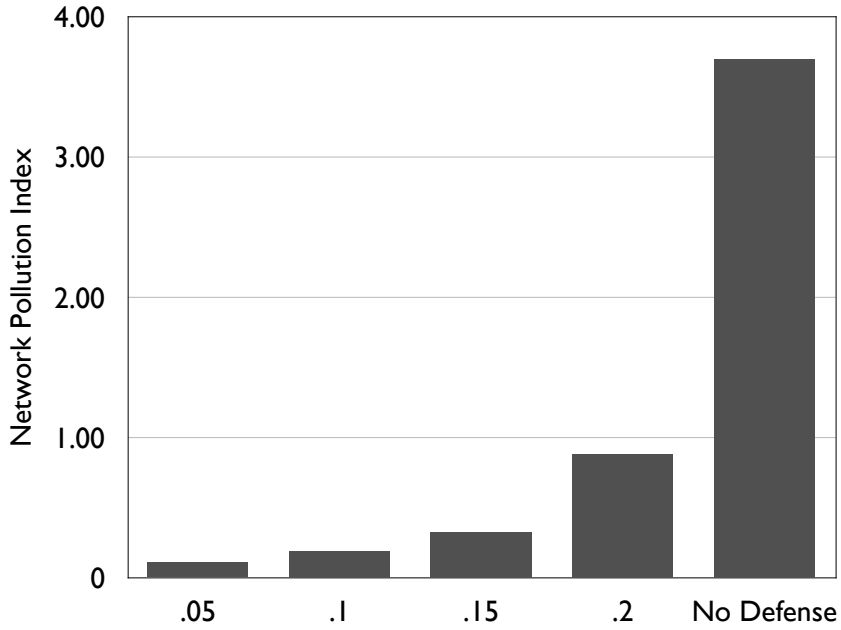
The multimedia content is streamed at a rate of $64KB/sec$, a typical rate for standard definition video. Each segment represents 1 second of the playback, *i.e.*, $64KB$ of video data. In order to study the effectiveness and the impact of the reputation system in P2P streaming systems, we must maximize the participation of the peers. Hence, the upload bandwidth of each peer is set to $95KB/sec$, more than enough to support the data stream and protocol-related control messages. The streaming server upload bandwidth is $1MB/sec$, just sufficient to support smooth playback in the system. The data swarming in the network is governed by a pull-based streaming protocol (similar to [58]), the most common approach found in commercial systems [3, 4]. In this type of data swarming, a peer actively searches for a missing segment among its neighbors and requests for it if any of the neighbors has it. To avoid overloading a peer with an excessive number of connections and requests, the number of upstream peers and downstream peers is limited to 10. The streaming server does not need any upstream peer. Since it has higher upload bandwidth, we limit the number of its downstream peers to 30. Each streaming session lasts 10 minutes. In the first 3 minutes, peers join the network following a Poisson distribution with $\lambda = 1.11$. In the next 4 minutes, all 199 peers form a stable network for the streaming session. In the last 3 minutes, peers leave the network following the same Poisson distribution. This setting allows us to observe both the static and the dynamic cases of the data swarming. As in real streaming systems, we implement an initial delay of 30 seconds. During this time, a peer does not play any segment and tries to download as many segments as possible. This ensures smooth playback once the playback is started.

5.2 Performance Metrics

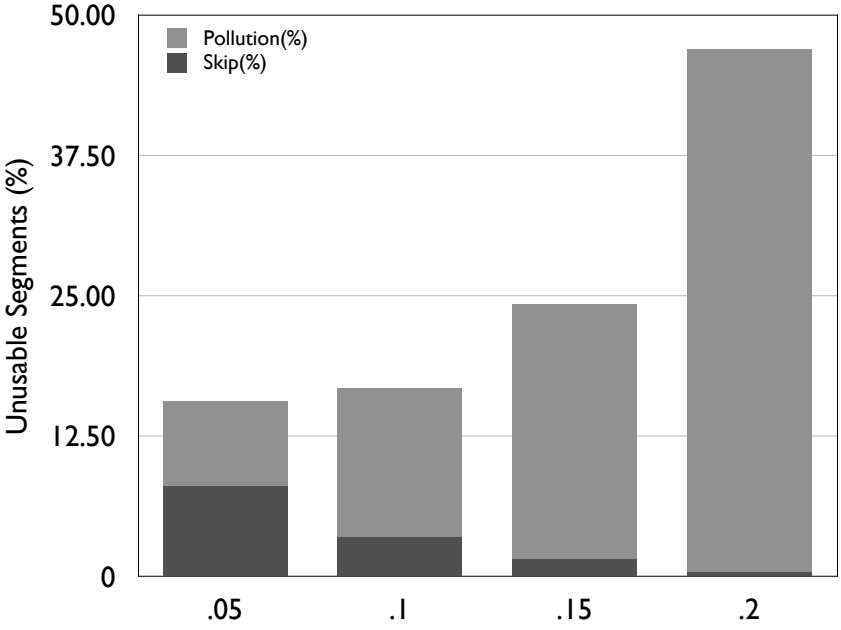
Besides the number of polluted segments, we also monitor two key performance metrics in our experiment. One is the network pollution index (NPI), which is the ratio of the number of polluted segments found in the network and the number of regular segments shared in the network. NPI is a good indicator for the severity of the pollution attack, as well as the effectiveness of the defense mechanisms. The other one is the playback quality. It is measured by the amount of segments that are skipped during a streaming session. We calculate this as the percentage of playback skips at the peers out of all segments to be played during the 10-minute session. With this setting, our streaming system achieves less than 0.05% of playback skips, i.e., less than 0.3 seconds are skipped during playback on average for each peer. Furthermore, we calculate the percentage of polluted segments that are played during the session.

5.3 The Global Reputation System

In the global reputation rating system, peers periodically report their observations about neighboring peers to the server, so that the global normalized reputation ratings can be computed for all peers in the network. The server recommends neighbors to peers based on the global reputation ratings. As described in Sec. 4.2, the recommendation can be made in one of two ways. The server suggests the next best ranked peer that still has room to take more downstream peers [32]. Alternatively, the server randomly picks a peer with reputation rating lower than a predefined threshold [15, 17, 36]. The threshold value defines the percentage of polluted content served by a peer among all the segments it has served, *e.g.*, 0.05 threshold value means at most 5% of misbehavior from a peer is allowed. When a peer's reputation is higher than the threshold, it is considered a polluter and will not be recommended by the server.



(a) NPI for different threshold values



(b) Unusable segments for different threshold values

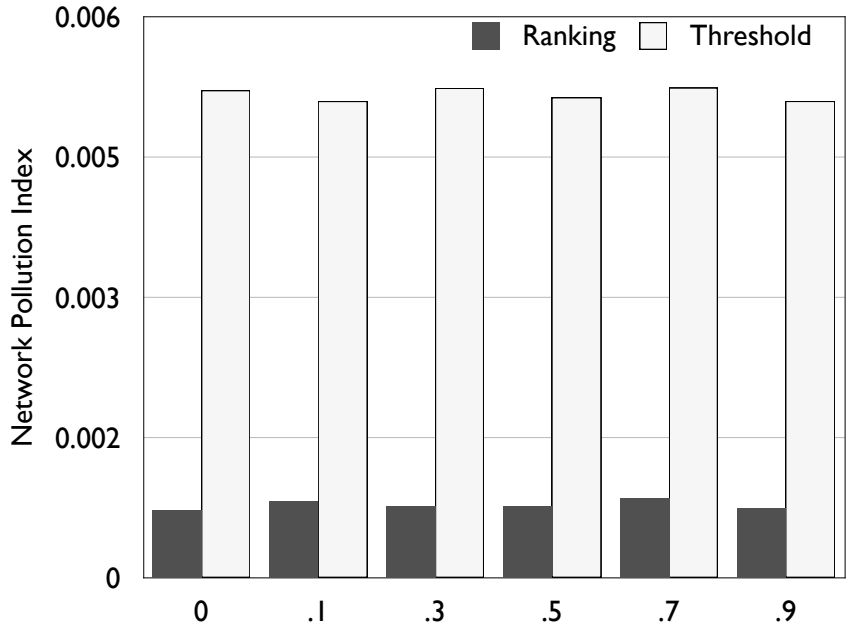
Figure 5.1: Tuning the threshold values for the global reputation system

In Fig. 5.1(a), as we vary the threshold value from 0.05 to 0.2, the NPI increases linearly, but still achieves at least 4 times improvement compared to the case where no defense mechanism is in place. For loose threshold, almost 50% segments in the network are unusable. This is a logical amount under heavy pollution attacks [6]. A tight threshold helps to pin-point the polluter. However, the tighter the threshold is, more innocent peers are likely to be filtered out as polluters, which reduces the bandwidth supply in the network, leading to increased playback skips as shown in Fig. 5.1(b). In fact, two opposite trends are visible among the playback skips and percentage of polluted segments played. As opposed to the playback skips, the percentage of polluted segments follows the increasing trend of NPI with the increase of global threshold. With loose threshold, fewer peers are avoided by the server for selection, which increases segment availability, but allows more polluters to get downstream peers. The overall playback quality improves with tighter threshold values and becomes steady from 0.1 and lower values. For the same overall playback quality, we prefer fewer playback skips and want to reserve buffer spaces for false alarms on innocent peers. Therefore, we fix the threshold at 0.1 for the rest of the experiments.

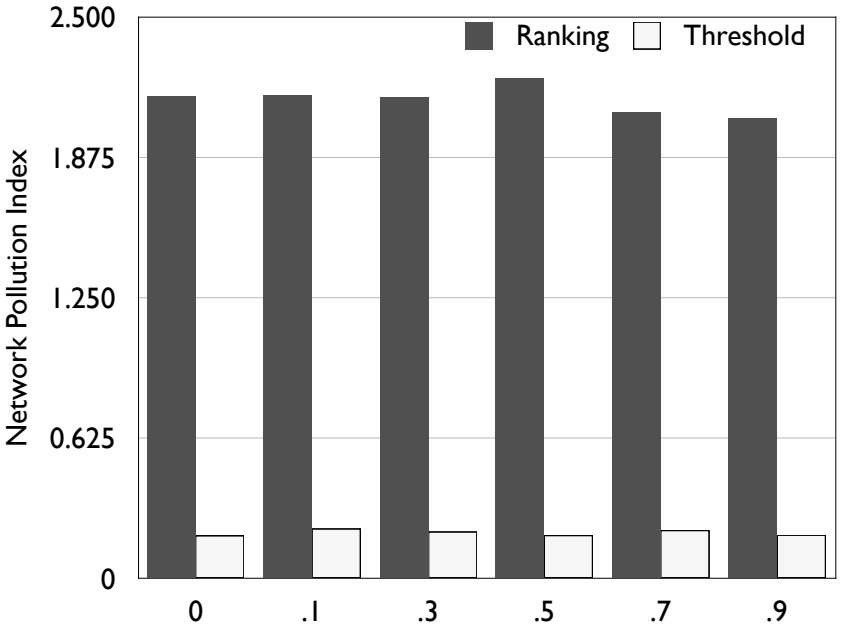
Table 5.1: Playback quality for different values of ϵ

	Skip(%), Pollution(%)	
ϵ	Global(Ranking)	Global(Threshold)
0.0	0.35, 68.51	3.50, 13.22
0.2	0.33, 68.38	3.12, 13.30
0.4	0.36, 68.35	3.85, 13.83
0.6	0.34, 68.13	3.14, 13.29
0.8	0.35, 68.83	3.39, 14.06
1.0	0.35, 67.93	3.34, 13.47

Next, we examine the weighting factor ϵ (from Eqn. 4.8) used in calculating the global reputation of a peer. Here, we conduct the experiments with only a single polluter



(a) NPI from a single polluter (light attack)



(b) NPI from 10% polluter network (heavy attack)

Figure 5.2: NPI of the global reputation system when tuning ϵ

and 10% polluters, representing a light pollution attack and a heavy pollution attack, respectively. According to Fig. 5.2, the choice of the ϵ value has no impact on the severity of the pollution attack, regardless of the implementation choice on the server and the intensiveness of the attack. The playback quality under a heavy pollution attack, as in Table 5.1, also shows no significant difference for different values of ϵ. This is because that a polluter in the default pollution model does not change its behavior throughout the entire session. Hence, the reputation from the past is likely be the same as the new reputation. As a result, we drop the weighting factor ϵ, and update Eqn. 4.8 to the following:

$$G_i = Avg_i = \sum_{j=1, j\neq i}^{N} G_j r_{ij} \tag{5.1}$$

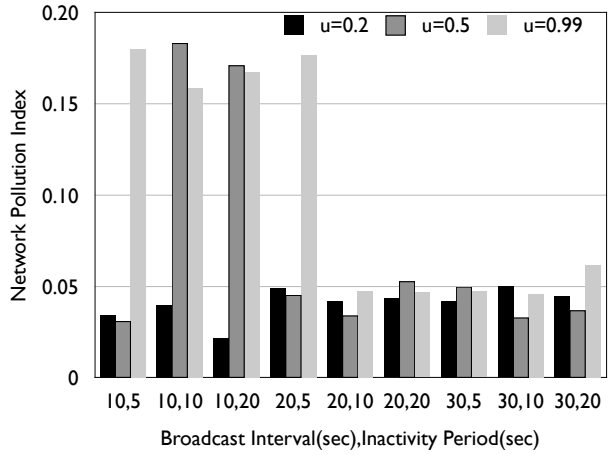
One interesting observation from Fig. 5.2 is that under a light attack, the ranking approach performs better than the threshold approach, and vice versa when the network is under a heavy attack. This is further confirmed by Fig. 5.5(a) in Sec. 5.5, from which we note that, the ranking approach starts to lose control over the pollution even with only 4% polluters. We shall discuss this observation in Sec. 5.5.

5.4 The Local Reputation Rating System

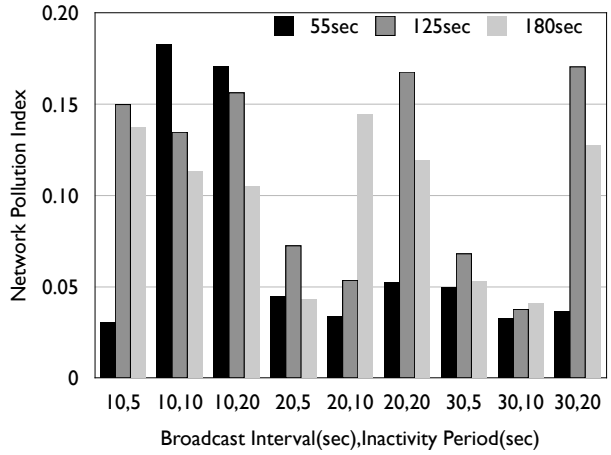
Among the peers in a P2P streaming system, peers exchange control messages that govern the streaming session and share received content with each other. Such a cooperative infrastructure can also facilitate the reputation rating system described in Sec. 4.1. In this chapter, we examine two variations of the system: one is based only on first-hand information, and one is based on both the first-hand and second-hand information. We note that the first variation includes the global system studied in Sec. 5.3, where the first-hand information is sent to the server to calculate the global reputation values. In

this section, we tune the reputation management parameters for the second variation - the fully distributed rating system.

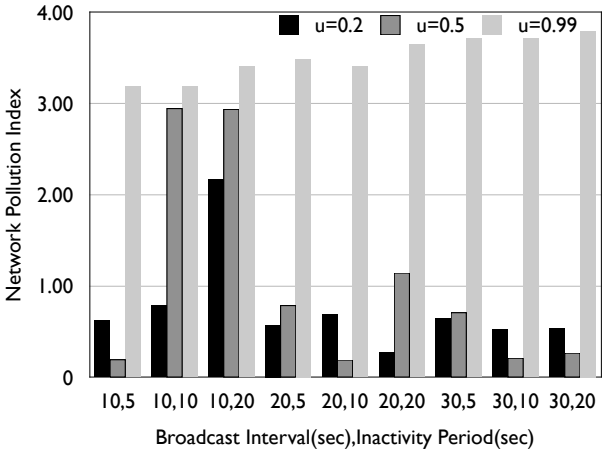
We begin the study with the trade-off found among three key parameters, namely the broadcast interval, inactivity period and u, for controlling the reputation rating system. Here u determines how much previous ratings should be included in the new one. Fig. 5.3 shows the NPI under both the light pollution attack (Fig. 5.3(a) and Fig. 5.3(b)) and the heavy pollution attack (Fig. 5.3(c)). We choose several representative settings for the broadcast interval of the reputation sharing and the predefined inactive period. As illustrated in Fig. 5.3(a), it is rather counter-intuitive that the short broadcast intervals lead to higher NPI, especially when long history (large u value) is kept in the rating. Though frequent broadcast of the rating quickly disseminates any misbehaving information across the network, according to Eqn. 4.3, each newly received rating will lead to an increase in the reputation rating parameters. Moreover, this problem is more severe when the polluters join the network late. The reason is, if a polluter joins the network later, it has more upstream peers than the downstream peers. Since a peer initially assigns a neutral reputation rating (which is less than the reputation threshold) to each of its neighbors (both upstream peers and downstream peers), and only the downstream peers can provide negative feedbacks, the rating at a peer may boost the polluters' reputation and allow them to pollute more. Such a situation can be avoided by either increasing the broadcast interval or reducing the inactive period. In the latter case, the reputation parameters (α and β) of a peer will decrease rapidly when there is no active traffic flowing from a neighbor peer. Thus, a small amount of first-hand observations will have a large impact on the reputation ratings of a peer. From Fig. 5.3(b) and Fig. 5.3(c), we note that the system will work regardless of the join time of the polluters and the intensity of the pollution with the settings of a 30-second broadcast interval, 10-second inactive period, and $u = 0.5$. From here on, we shall use this setting.



(a) with 1 polluter network (light pollution)



(b) with 1 polluter network and $u = 0.5$ (light pollution)



(c) with 10% polluters (heavy pollutions)

Figure 5.3: Tuning broadcast interval, inactivity period and u in Eqn. 4.1

Table 5.2: Impacts (NPI) of the second-hand information and the thresholds

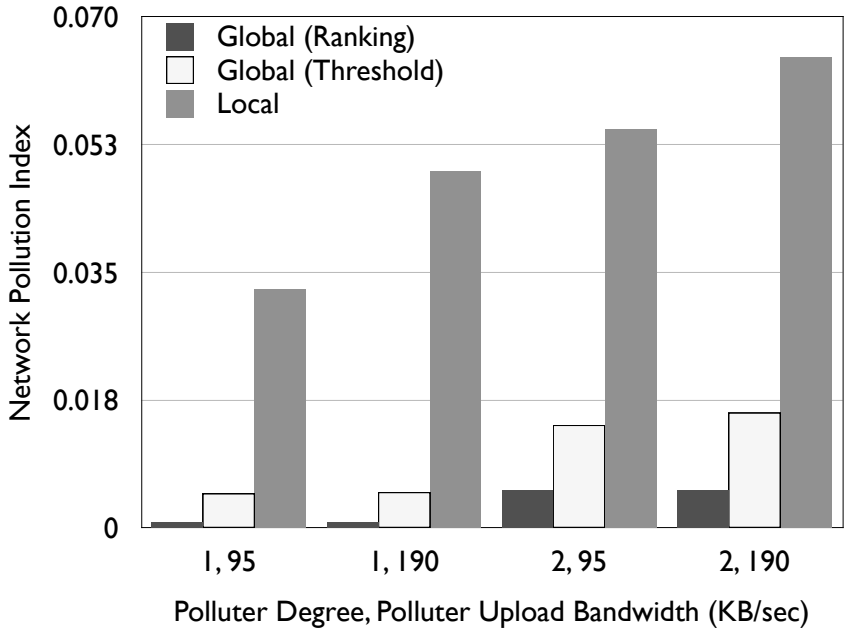
	v	r	t	w
0	0.22	—	***	0.5
0.1	—	—	0.55	0.21
0.15	—	—	—	0.27
0.2	—	—	0.27	3.47
0.25	—	—	—	***
0.3	0.21	—	0.23	***
0.5	0.22	—	—	***
0.55	—	0.20	0.22	***
0.6	—	0.21	0.21	***
0.65	—	0.28	0.21	***
0.7	0.21	0.33	0.22	***
0.9	0.22	***	—	***

Table 5.3: Impacts (playback quality) of the second-hand information and the thresholds

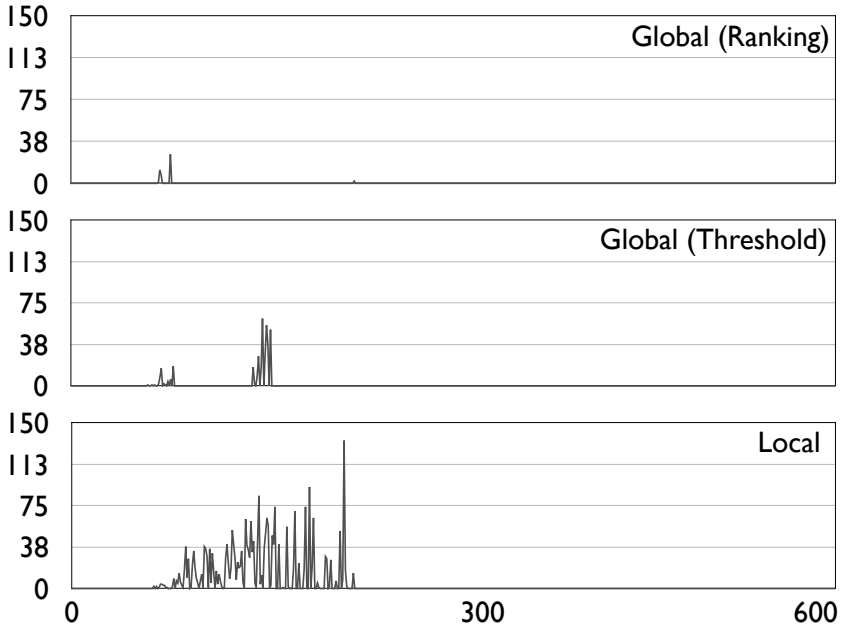
	Skip(%), Pollution(%)			
	v	r	t	w
0	0.12, 18.17	***	***	2.92, 27.38
0.1	—	***	16.53, 27.52	0.14, 17.43
0.15	—	***	***	0.11, 20.96
0.2	—	***	1.55, 20.91	0.06, 70.62
0.25	—	***	***	***
0.3	0.14, 17.43	***	0.24, 18.69	***
0.5	0.16, 17.92	***	—	***
0.55	—	5.08, 15.54	0.15, 18.10	***
0.6	—	0.14, 17.43	0.14, 17.43	***
0.65	—	0.14, 21.93	0.21, 18.10	***
0.7	0.16, 17.02	0.21, 24.85	0.24, 18.26	***
0.9	0.22, 17.20	—	—	***

The objective of the rating of a peer depends on the consensus of the entire neighborhood. Next, we study the impact of the second-hand information, controlled by v in Eqn. 4.4 and w in Eqn. 4.3. Furthermore, we varied the threshold for the reputation rating, r from Eqn. 4.6, and trust rating, t from Eqn. 4.7. Table 5.2 shows the NPI as Table 5.3 shows the playback quality as we tune each of these parameters from 0 to 0.9. The '—' entry indicates that such a setting leads to no significant change from the other entries and the '***' entry means that the setting leads to too many playback skips or high NPIs. For example, while tuning w in Table 5.2, we do not need to tune higher values than 0.2 since the results are already unacceptable. Moreover, we observe that the NPI decreases as reputation threshold r is set to lower values, and there is no significant change from $v = 0.5$ to $v = 0.7$.

Among the four parameters, the weighting factor for the second-hand information, w, has a more noticeable impact. As more second-hand information is incorporated, the system is more accurate in detecting and isolating the polluters. However, the cut-off point is $w = 0.2$, at which point, the NPI increases significantly, as shown in Table 5.2. The amount of polluted segments played, as in Table 5.3, also shows the same trend. We find that the right ratio is around $w = 0.1$, where both NPI and the percentage of played polluted segments are the lowest. We further find that the history factor v has little impact on the performance, since all polluters are behaving the same throughout the entire session. While tuning t, we find that there is a lower cut-off value, above which the NPI becomes stable. The playback quality in Table 5.3 agrees with this observation. Furthermore, the playback quality becomes acceptable for $r \geq 0.6$. We settle on the two threshold values $t = 0.6$ and $r = 0.6$ for the best balance.



(a) NPI



(b) # of polluted segments

Figure 5.4: The performance when varying the polluter degree and upload bandwidth, under a light pollution attack

5.5 Polluters' node degree and upload bandwidth

From Sec. 5.3 and Sec. 5.4, we learn that the most effective solution for lightly polluted networks is the global ranking system. For heavyly polluted networks, both the global threshold system and the local rating system are approximately equally effective. As concluded from [62], the node degree of a polluter is more important than the upload bandwidth of the polluter. The node degree a polluter here is the number of polluters present in the system. To confirm the theoretical conclusion, we examine the three systems under different degree-bandwidth settings. For light-pollution attacks, we begin with a single polluter with $95KB/sec$ upload bandwidth, and then double its bandwidth to $190KB/sec$. Next, we increase the number of polluters to 2, and reduce their upload bandwidth back to $95KB/sec$, so that the total bandwidth remains the same as the previous case. Finally, we double the upload bandwidth of both polluters. Fig. 5.4(a) shows the performance of different defense mechanisms under the four degree-bandwidth settings. It is clear that the global ranking approach produces the best results, and the local rating system produces the worst. With global knowledge, it is easy to identify and isolate the polluters. As shown in Fig. 5.4(b), the amount of polluted copies of each segment of the video is significantly higher in the local rating system. The time taken to detect and isolate the polluters is quicker in the global ranking system, since the polluters' ranks are bad even when their ratings are not above the threshold. Thus the polluters will stand out in light-polluted networks regardless of the status of the network and because almost all peers are behaving, a peer can find sufficient upstream peers without including the polluter. We also observe that for the same pollution bandwidth, 2 polluters with $95KB/sec$ upload bandwidth have greater impact than 1 polluter with $190KB/sec$ upload bandwidth.

In the case of heavy pollution, we again look at the four cases. Among 199 peers,

Table 5.4: Performance of defense mechanism for different polluter degree and upload bandwidth

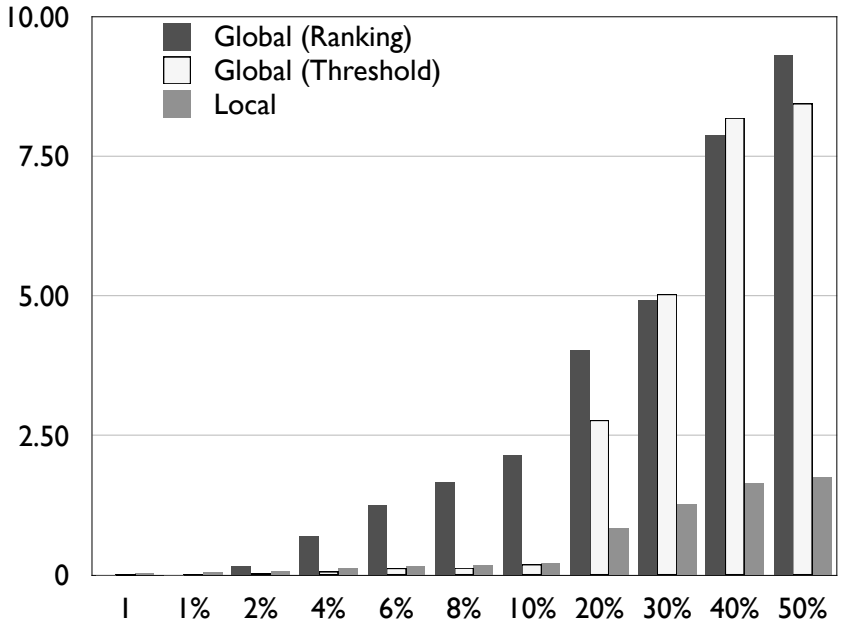
	Fixed bandwidth			Fixed node degree		
Polluters	G-R	G-T	L	G-R	G-T	L
10%	20 polluters $95KB/sec$			10 polluters $190KB/sec$		
	2.15	0.19	0.21	1.27	0.07	0.14
20%	40 polluters $95KB/sec$			10 polluters $380KB/sec$		
	4.99	5.92	1.08	1.63	0.07	0.13

G-R=Global(Ranking), G-R=Global(Threshold), L=Local

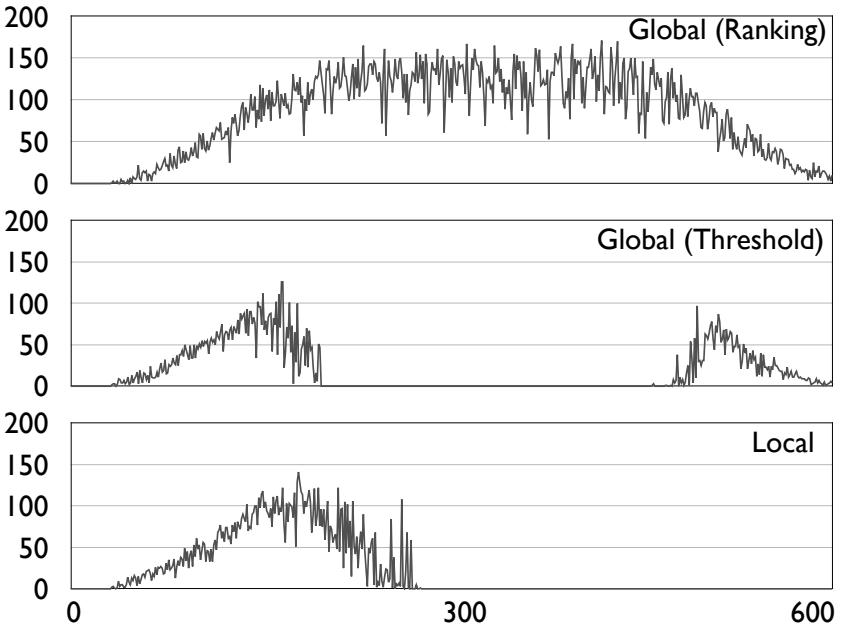
we first set 20 peers to be the polluters with $95KB/sec$ upload bandwidth each. We then reduce the number of polluters by half and double the upload bandwidth, so that the total bandwidth among the polluters is still the same. We repeat this for a larger number of polluters. The result is presented Table 5.4. It is clear that for all rating systems, performance decreases for larger node degree. In fact, increasing in polluters' upload bandwidth has very little impact.

Finally, we look at the effectiveness of the three systems under attacks launched by different number of polluters. We increase the number from 0.5% (a single polluter) to 50% of the network. We note that in all these cases, all polluters have the same upload bandwidth. Fig. 5.5(a) indicates that, as expected, the performance of all three defense mechanisms degrade as the pollution gets heavier in the system, but at different rate. The performance of the global ranking approach degrades linearly as the number of polluters increases. The global threshold system is quite effective up to with 10% polluters, and then collapses for heavier pollution attacks. Starting from the 30% polluter network, its performance becomes as bad as the global ranking approach. The local rating system is much more scalable, with a slight increase of NPI for cases with more than 20% polluters. Still, it performs at least 50% better than the global approaches in these cases.

The interesting observation from Fig. 5.5(a) leads us to conduct a more detail anal-



(a) NPI



(b) # of polluted segments

Figure 5.5: Performance under increasing pollution attacks

ysis. In Fig. 5.5(b), we compare the number of polluted copies for each segment in the 10-minute session. Here, polluters tend to be chosen as upstream peers, because the normalized rating of all peers are close to each other. Innocent peers surrounded by many polluters may be ranked even worse than a polluter, *i.e.*, there are more opportunities for the polluters. Even if the polluters stay at the bottom of the ranked list, since the global ranking system only ranks peers and does not exclude them in peer selection, polluters may still be selected when the bandwidth among the regular peers is running out, especially when the number of polluters is large among the peers. Consequently, as the number of polluters increases, some polluters can have many downstream peers even if they are actively polluting the network. This leads to a smoother playback among peers (in terms of playback skips), although a good number of the played segments are polluted. According to the Fig. 5.5(b), the pollution goes on during the entire streaming session, which is the result of not excluding any misbehaving peers. We note that the cause of the number of polluted segments being less at the beginning and the end is that those are the peer joining and leaving periods and fewer peers are present in the system during those periods. Both the global threshold and the local rating approaches cut polluters off according to the predefined threshold values, *i.e.,* once the polluter is detected, it is isolated and is expelled from the neighborhood. Although the NPIs for the two approaches are the same, the distributions of the polluted copies are very different. Since the global threshold approach uses global ratings, the detection process is faster, which makes the spikes in Fig. 5.5(b) end around 200 seconds into the session. Still, it produces similar results as the local approach, because it normalizes the ratings, thus, when there are fewer peers in the network, some of the polluters will have good ratings. That's why there are more polluted contents not only at the beginning, but also at the end of a session. In contrast, the local rating method allows each peer to maintain its own rating system and no normalization is applied. Although it detects the polluters more

slowly, once a polluter is detected, it is avoided by the peers around the neighborhood. For this reason, its performance is the best under heavy pollution attacks.

5.6 Polluter join time

In this section, we study the impact of the join time of the polluters. Instead of letting the polluters join the session following a Poisson distribution, we let all polluters join at the same time. Such a flash-crowd scenario among the polluters has the potential to affect the severity of the pollution and the playback quality. Fig. 5.6 shows that, since the global ranking approach does not expel the polluters, it cannot effectively reduce the NPI in the system. However, as the network becomes more established, the polluters are ranked bad immediately after joining the network, as indicated by the decreasing trend in Fig. 5.6. According to the figure, in terms of NPI, both the global threshold and the local rating approaches are equally effective, and are immune to the flash-crowd scenarios.

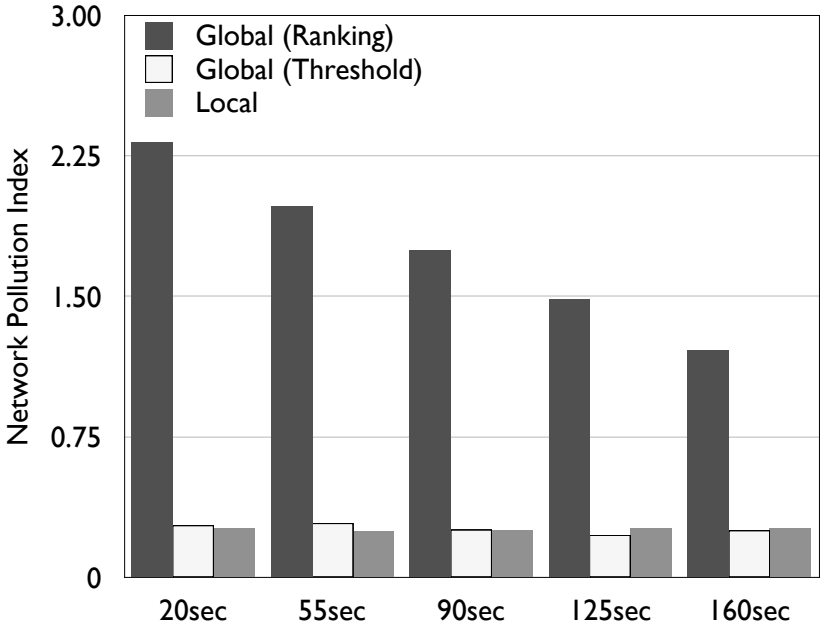


Figure 5.6: NPI for different polluter join times

Table 5.5: Playback quality for different polluter join times

	Skip(%), Pollution(%)		
Polluter Join Time	Global(Ranking)	Global(Threshold)	Local
20sec	0.26, 64.11	1.93, 18.99	0.37, 20.68
55sec	0.26, 61.07	0.41, 20.07	0.92, 19.56
90sec	0.27, 58.23	0.15, 18.72	2.05, 19.57
125sec	0.28, 54.64	0.13, 17.53	4.38, 19.76
160sec	0.29, 49.52	0.16, 18.58	5.64, 19.31

Unlike the global ranking approach, the global threshold and the local rating approaches cut-off suspicious peers. The loss in bandwidth supply due to the cut-offs leads to poor playback quality. From Table 5.5, among the two parameters of playback quality, the percentage of skips is persistent, since no peers are excluded in the data swarming. The percentage of played polluted segments reflects the effectiveness of the three defense mechanisms. We note that the global threshold approach is less effective when polluters join the network earlier. When polluters are expelled from the system, peers that are relying on the polluters will experience a loss of bandwidth, *i.e.*, higher skip percentage. As the streaming session progresses, the reputations of the existing peers are already established, and any new peer, either a polluter or a regular peer, will have bad rating, especially after normalization. Therefore, it is harder for the polluters to boost its rating if they join the network later.

With the local rating approach, since all peers operate independently of each other, newly joined peers will be rated bad only after receiving polluted content from them. When they are detected and removed from the system, all peers in the neighborhood will experience a shortage of bandwidth before they find new valid neighbors. Thus, the amount of skips in this case increases with the change of polluter join times, while the percentage of played polluted segments becomes persistent and is similar to that of the global threshold approach.

5.7 Whitewashing Attacks

In this section, we examine the three defense mechanisms against a more challenging attack, the whitewashing attack. As we described in Chapter 3, such an attack emerges to counteract the reputation-based rating systems. A polluter may leave and then join the network as a new peer to erase its bad reputation. A polluter may also mix the good content and polluted content in order to keep its ratings below the threshold. To simulate this behavior, we configure the polluters to send polluted segments with a certain probability (P). The lower values of P allow the polluters to earn better rating. When this attack is launched in association with heavy pollution attack, we observe different performance with different defense techniques, as in Fig. 5.7. Since under heavy attack, some polluters always rank good in the global ranking approach and misbehaving peers are never expelled, some polluters can always send polluted segments, although fewer polluted segments are found in the system.

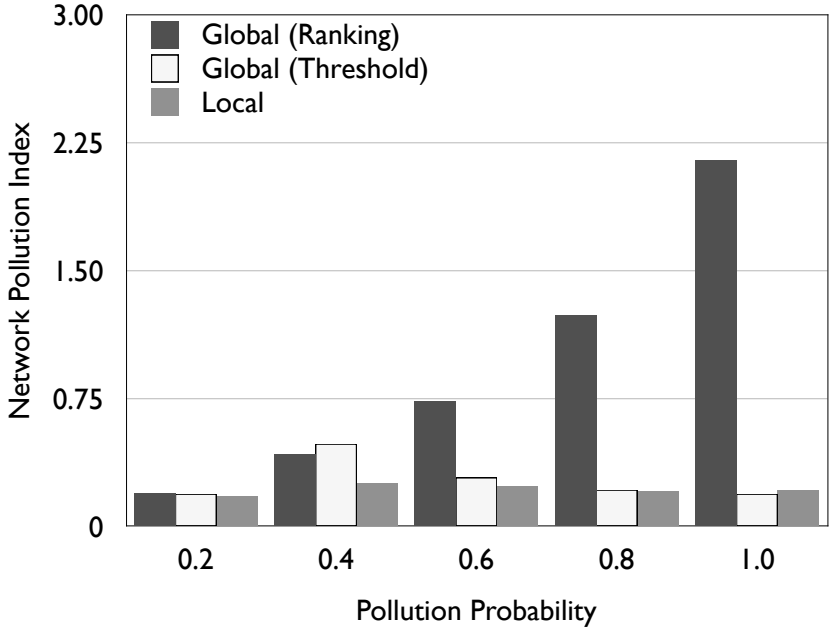


Figure 5.7: NPI for different pollution probability

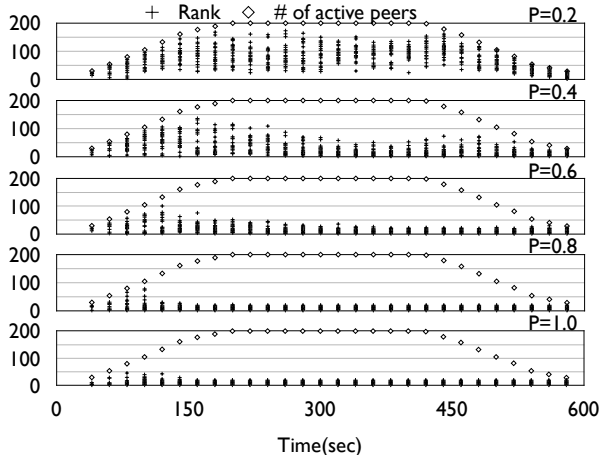
In fact, the threshold-based approaches are affected the most by the whitewashing

attack. A polluter can hide in the system by keeping its reputation lower than the threshold. Table 5.6 shows that, for the global threshold approach, the impact on the playback quality by the whitewashing attack rises to the peak at $P = 0.4$ and gradually decreases for higher values. With the local approach, although NPI does not change significantly as in Fig. 5.7, the playback quality follows the same trend as the global threshold approach. With higher probabilities, the pollution is easy to detect. While observing the playback skips in the global threshold approach, with higher pollution probabilities, polluters' reputation rating drops below the threshold value due to the large amount of polluted content injected into the network. As they are being expelled from the session, it takes peers time to recover from the instantaneous loss of bandwidth supply. The local rating system is the most effective solution here, since it offers low NPI while maintaining relatively better playback quality. Compared to the global threshold approach, the local rating system isolates the polluters over time in a distributed manner, so there is no sudden loss of bandwidth supply.

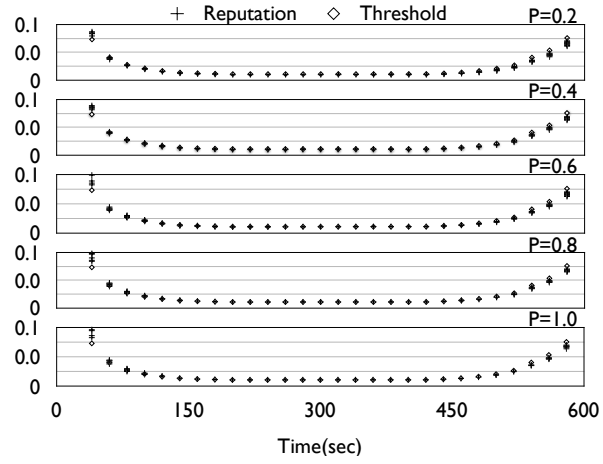
Table 5.6: Playback quality for different pollution probabilities

	Skip(%), Pollution(%)		
Pollution Probability (P)	Global(Ranking)	Global(Threshold)	Local
0.2	0.25, 15.48	0.12, 15.07	0.61, 14.49
0.4	0.32, 27.43	0.26, 30.16	1.32, 19.75
0.6	0.26, 38.99	0.16, 20.29	0.34, 19.20
0.8	0.3, 50.93	1.98, 15.01	0.14, 17.26
1.0	0.3, 62.68	3.50, 13.22	0.16, 17.58

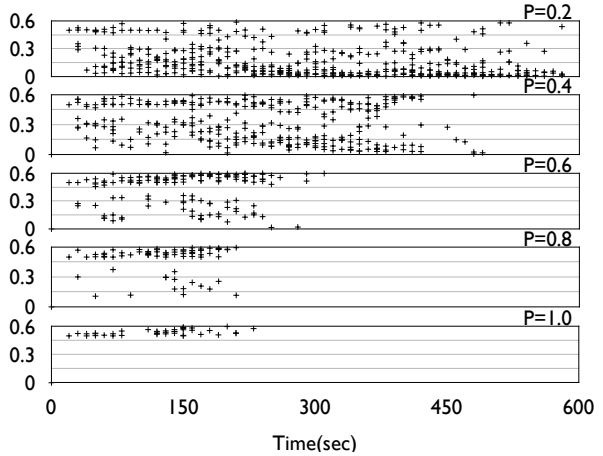
Next, we study the polluters' ranks for different pollution probabilities over time. In Fig. 5.8(a), we show the ranking of all polluters (up to 20 polluters) and the number of peers in the network over time. Due to the normalization performed on the server, all polluters receive acceptable rank when the pollution probability is low. In Fig. 5.8(b), we observe that due to the normalization of the global reputation ratings, the polluters'



(a) Global (Ranking)



(b) Global (Threshold)



(c) Local

Figure 5.8: Change of reputation with time for different pollution probabilities

reputations become close to the threshold value. Hence, the polluters whose ratings are near the threshold values can easily become active with a small amount of good transactions. For the local rating system, we collect the rating of a particular polluter from its neighbors and show how the ratings from neighbors change over time in Fig. 5.8(c). With larger pollution probability, the polluters can act as upstream peers for only a very short period of time. However, the NPI remains unchanged in Fig. 5.7 regardless of the pollution probability. This means that the total amount of polluted content allowed by the local rating system remains the same. The polluters either inject a large amount at the beginning or constantly inject a small amount throughout the session.

5.8 Impact of collusion attack

The collusion attack is another countermeasure against modern reputation-based defense mechanisms. In this type of attacks, the polluters collaboratively rate each other very good. They can also rate the regular peers bad to confuse the rest of the system. As described in Chapter 3, to examine its impact, we simulated three collusion scenarios: false positive (polluters rate each other good), false negative (polluters rate innocent peers bad), and both. Fig. 5.9 shows the impact of collusion with the presence of the three defense techniques. There are three key observations: (1) the global approaches are more susceptible to false positive than the local defense; (2) threshold-based approaches perform better; and (3) the local rating approach is immune to the collusion attacks. The global approaches do not have any mechanism for verifying the trustworthiness of the reported reputation values. The server simply believes the reporting peers and directly accepts the reported reputation values without any verification. In contrast, the local rating system has a separate trust module similar to its reputation module that is used to filter the incoming second-hand reports. It strictly verifies the trustworthiness of the

reporters and incorporates the reports only if the reporter passes the trustworthiness test.

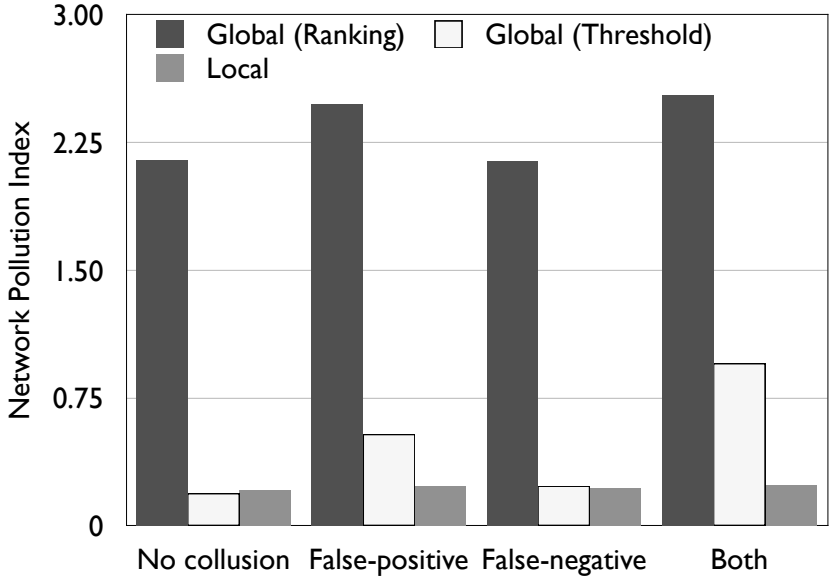


Figure 5.9: NPI under different collusion scenarios

Table 5.7: Performance (NPI) for different global defense parameters

	Ranking		Threshold	
Parameter values	ϵ	u	ϵ	u
0.0	2.48	2.75	0.95	3.59
0.1	2.45	2.59	0.94	1.39
0.3	2.46	2.48	0.96	0.94
0.5	2.48	2.48	0.95	0.95
0.7	2.47	2.26	0.96	2.03
0.9	2.46	2.45	0.94	2.35

We now examine the effects of different parameters on the defense mechanisms. Table 5.7 presents the results from tuning parameters for the global rating systems. The ranking system is much less effective in terms of NPI and playback quality. Consistent with our previous observations, none of the ranking and threshold approaches rely on the value of ϵ in the global reputation calculation. Tuning the history parameter u has

Table 5.8: Performance (playback quality) for different global defense parameters

	Skip(%), Pollution(%)			
	Ranking		Threshold	
Parameter values	ϵ	u	ϵ	u
0.0	0.32, 71.48	0.37, 71.07	0.32, 47.57	0.43, 77.98
0.1	0.38, 70.45	0.38, 70.95	0.30, 47.78	0.41, 55.87
0.3	0.38, 70.81	0.38, 71.07	0.35, 48.45	0.35, 47.55
0.5	0.36, 71.05	0.32, 71.48	0.31, 47.12	0.32, 47.57
0.7	0.33, 70.98	0.37, 70.07	0.35, 48.09	0.94, 65.88
0.9	0.37, 70.81	0.35, 70.58	0.34, 48.04	4.50, 63.81

no significant impact on the ranking approach. In the threshold-based approach, both NPI and playback quality decrease when u is assigned values beyond a certain range. According to Table 5.7 and Table 5.8, the best performance can achieved with the range $0.1 < u < 0.7$.

Table 5.9: Performance (NPI) for different local defense parameters

Parameter values	v	t	w
0.0	0.22	—	0.5
0.1	—	—	0.23
0.2	—	0.24	3.28
0.3	0.23	0.21	***
0.4	0.23	0.22	***
0.5	1.30	0.23	***
0.6	***	0.23	***
0.7	***	1.17	***

For the local rating system, we believe that the parameters for the second-hand information are important in fighting against collusion attacks. The local second-hand parameters v, t and w are the ones controlling the consideration of second-hand information. According to Table 5.9 and Table 5.10, keeping less than 50% trust history, v, can mitigate the effects of collusion effectively, since neighbors of polluters always receive

Table 5.10: Performance (playback quality) for different local defense parameters

	Skip(%), Pollution(%)		
Parameter values	v	t	w
0.0	0.20, 19.55	***	2.92, 27.38
0.1	—	***	0.21, 20.72
0.2	—	0.63, 20.79	0.18, 76.50
0.3	0.21, 20.72	0.16, 19.30	***
0.4	0.22, 20.87	0.17, 19.40	***
0.5	66.15, 20.07	0.16, 19.38	***
0.6	***	0.21, 20.72	***
0.7	***	65.33, 19.30	***

false reports. Less history helps to omit reports from the polluters. Table 5.9 also shows that a trust threshold, t, tighter than 0.7 minimizes NPI. Playback quality in Table 5.10 also improves within this range. Similar to our previous findings, a small amount of second-hand report ($w = 0.1$) is enough to reduce the attack.

5.9 Summary

In this Chapter, we presented a deep view on the global and local reputation-based approaches. Our findings are as follows:

- The simulated defense techniques confirm the theory that node degree is more effective than bandwidth.
- The centralized global reputation-based approaches can create the total view of the network faster than the local approach.
- The global approaches perform better in light pollution scenario, because they use simple comparison among the reputation ratings of the peers (ranking or normalization), which can detect slight change of ratings and thus isolate the polluters.

- Under heavy pollution attacks, due to the large number of polluters, and the fact that heavy dissemination of polluted contents can affect the reputation of regular peers, the global ranking approach collapses.
- The threshold approaches, because of the cut-off of peers based on their reputation ratings, perform better under any heavy pollution.
- For the same reason, the threshold approach can cope with the flash crowd scenario to reduce pollution. The local approach, however, due to the lack of global information, suffers from sudden shortage of bandwidth in the neighborhoods, causing high playback skips for later arriving polluters.
- Although whitewashing attack can degrade performance of the threshold-based approaches, the local approach can keep pollution persistent and the amount of pollution here is lower.
- Because of the lack of trust management, the global approaches cannot cope against collusion attacks.

Chapter 6

DRank: A Distributed Rank-based Reputation Rating System

Our findings in Chapter 5 demonstrate that the global ranking approach performs the best in terms of NPI under light pollution attacks. With global information about the peers, the server can quickly rank the polluters bad, before they can cause too much damage. Under heavy pollution attacks, with more than 4% polluters in the network, however, the effectiveness of the global ranking approach is very limited. Since there is no threshold to cut-off the polluters, it allows the polluters to continuously pollute the system and cannot distinguish polluters from their immediate neighbors. The local reputation approach is very effective in this case. With the local approach, the peers combine their own observations and suggestions from others in the network to detect polluters. A reputation threshold is used to cut off suspicious peers from the data swarm. As a result, they can identify any number of polluters after several rounds of rating updates. However, since this identification takes some time, and the peers do not have global information about other peers, its performance under light pollution is not as good as the global ranking.

The pros and cons of the global and local reputation ratings systems inspire us to explore the possibilities of combining the approaches. Since the global ranking and local approaches perform better than the global threshold approach in light and heavy pollution attacks, respectively, we choose to combine the ranking approach with the local approach. In this chapter, we present two combinations: GRank and DRank. GRank simply combines the global reputation ranking and local observations, where DRank incorporates the ranking system into local observations so that each peer ranks its neighbors independently.

Table 6.1: Performance comparison among different defense techniques in a 1-polluter network

	Broadcast only first-hand			Send broadcast interval ratings		
Defense	NPI	Skip(%)	Pollution(%)	NPI	Skip(%)	Pollution(%)
Global (Ranking)	0.0007	0.019	0.066	N/A	N/A	N/A
Global (Threshold)	0.005	0.033	0.56	N/A	N/A	N/A
Local	0.032	0.089	4.37	0.03	0.07	3.14
GRank	0.0005	0.019	0.05	0.0006	0.023	0.06
DRank	0.019	0.056	2.08	0.017	0.086	2.77

6.1 GRank: A Global Rank-Based Reputation System

GRank, our global rank-based reputation system, has two separate modules of defense. On the global side, similar to the global ranking technique, the bootstrap server maintains the global ratings of the peers, ranks them and suggests upstream peers according to the ranks. On the local side, in addition to simply sending the first-hand observation to the server, the peers broadcast their observations to the neighbors. Similar to the local approach described in Chapter 4, these observations are incorporated into the local reputation management. The peers send the ratings to the server. Based on the ratings, a peer can also remove neighbors whose ratings have dropped below a certain threshold. Unlike the local reputation system, peers in GRank will receive suggestions from the server based on the global ranking. This approach supports the peers to connect to the best reputable peers based on global knowledge, and at the same time, provides individual peers with control in choosing neighbors.

Table 6.1 and Table 6.2 present the performance achieved by different defense techniques under the default setting with light and heavy pollution attacks, respectively. NPI from the tables support the intuition that combining the global ranking and local rating offers a superb performance. According to Table 6.1, in the light pollution scenario, the performance of GRank in terms of NPI is similar to that of global ranking, and even better in terms of skips and percentage of pollution among the played segments. For

Table 6.2: Performance comparison among different defense techniques in 10% polluter network

	Broadcast only first-hand			Send broadcast interval ratings		
Defense	NPI	Skip(%)	Pollution(%)	NPI	Skip(%)	Pollution(%)
Global (Ranking)	2.15	0.32	62.04	—	—	—
Global (Threshold)	0.19	3.50	12.59	—	—	—
Local	0.14	0.55	13.21	0.15	0.92	13.11
GRank	0.14	3.61	12.01	0.14	1.89	10.70
DRank	0.073	0.12	7.36	0.072	0.64	7.22

10% polluter case, however, GRank causes higher playback skips, as shown Table 6.2. Because of the consideration of global knowledge, the polluters are ranked bad just after the first few transactions. Since the server tends to suggest the top ranked peers, the early joining innocent peers form neighborhoods among themselves. The polluters, because of their bad ranking, are left alone. Later joining peers, because of the scarcity of connections from early joined good peers, connect to the polluters. Since each peer can use its own judgement, most of these peers start to reject neighbor suggestions without connecting. As a result, the neighborhoods formed later are less stable than the early ones, which causes more skips. Other than that, the performance of GRank in terms of polluter detection is as effective as the local rating system for heavy pollution attacks.

6.2 DRank: A Distributed Rank-Based Reputation System

GRank, although able to perform better, is not flexible enough to meet the highly dynamic nature of the P2P streaming systems because it relies on the centrally computed global ranks. Furthermore, maintenance of the centralized rating system is costly in a P2P network, especially when the network is very large. This centralized feature is susceptible to any kind of attack that can be launched against any centralized system.

The dynamic and ever-changing nature of P2P streaming systems needs a fully distributed defense technique. Retaining the benefits of global ranking, but not limited by

the centralized computation, we propose DRank - a distributed rank-based reputation system. The main principle of this approach is to enable a peer to independently rate and rank the neighbors and to independently choose upstream peers. The local collection and accumulation of the reputation parameters is done similarly as the local rating system described in Chapter 4. In addition, we put a cap on α and β to prevent a polluter from accumulating good reputation in order to stay in the system longer when polluting the system. This is a typical strategy used in whitewashing attacks.

In DRank, we use the same two-layer security as in GRank, except that we bring the ranking solution to each peer in a fully distributed manner. It treats each peer as a fully independent entity to accumulate information and make decisions. Since the peers rank their known peers based on the gathered information, this approach enables the peers to be part of the best possible neighborhood subject to their own judgement. Thus, they not only can remove misbehaving peers, but also can carefully choose their source of segments. It has greater potential to fight against pollution attacks in P2P streaming systems. Moreover, because each peer ranks other peers locally, the peer ranking is not the same at all peers. So the choices of upstream peers are not uniform throughout the network. This makes the distribution of the polluters almost even in the network. This has the potential to improve the playback quality.

After joining the network, the peers periodically send connect requests to peers that are more reputable than the currently connected ones (Fig. 6.1). The decision to connect is based on the locally stored reputation values at each peer. This means that the reputable peers can be different from one peer to another. The peers rank all known hosts based on their reputation values calculated from their own observations and second-hand information from other peers, and connect to random set of peers. As time passes and a peer discovers more about the network, if a peers finds and successfully connects to a better ranked upstream peer, it will replace the worst ranked upstream peer (by

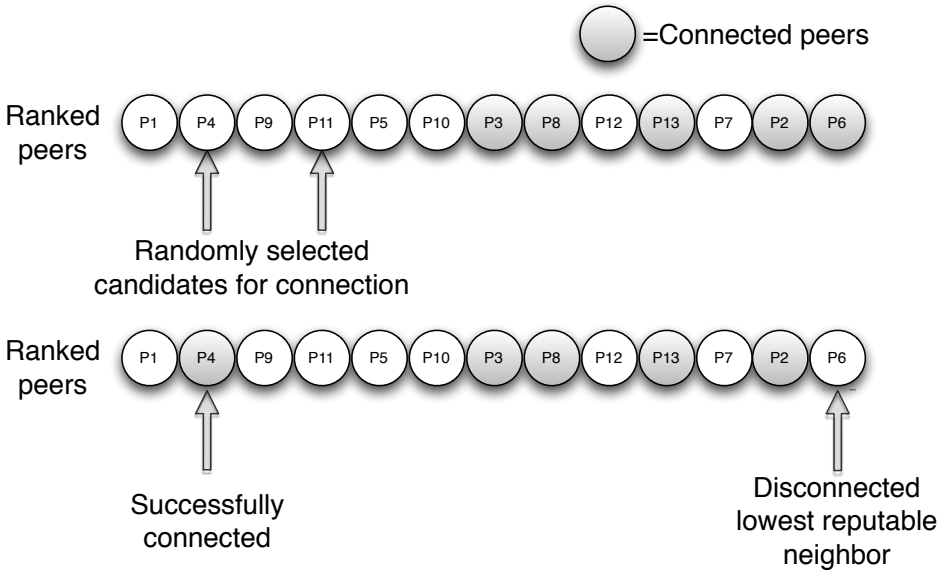


Figure 6.1: Process of connecting to better rated peers

disconnecting from it) with the new upstream peer.

We showed in Chapter 5 that the performance of the local reputation-based technique is highly dependent on the perfect tuning of the broadcast interval and inactivity period. Since the reputation value collection of DRank is done in the same fashion as the local approach, it is also subject to this tuning. We find from experiments that if we perform selective broadcast, we can avoid this tuning. Hence, each peer broadcasts ratings of only those peers that it is or was connected with to transmit segments. This modification also leads to better performance in terms of NPI. The reason is, in this case, our peers can broadcast local ratings more frequently, thus, can disseminate polluter behavior more promptly. We want to compare DRank with other approaches to see if the new idea to reduce pollution performs better. So for consistency of experimental results, we perform additional experiments of the local and combined approaches using the new broadcasting technique. For this reason, the experimental results for the local rating system in this chapter are slightly different from those in Chapter 5. Since the global approaches do not require peers to broadcast their local ratings, we do not need to repeat those experiments.

DRank does not have any centralized global rating facility, rather, its local ranking feature moves the peers to better neighborhoods slowly. Hence, it cannot outperform the global approaches under light attacks as shown in Table 6.1. Still, it has improved performance compared to the local rating system. We found in Chapter 5 that the local rating system performs the best under heavy pollution attacks. The local ranking feature of DRank provides extra advantage in addition to the local ratings, which helps DRank to outperform the local rating technique in this scenario, as shown in Table 6.2. Thus, under heavy pollution attacks, DRank performs the best.

While broadcasting, we can consider sending either 1) the ratings of all peers from which there was at least one segment received in the entire session, or 2) the ratings of peers whose first-hand ratings were updated after the last broadcast. Table 6.2 shows that the performance with the first technique is better in terms of skips and pollution for local and DRank. So we use this technique in the rest of the experiments.

In this chapter, in addition to the extensive measurement studies of DRank, as in Chapter 5, we perform a comparison study of this approach with the other defense approaches. We concluded in the last chapter and presented in Table 6.1 and Table 6.2 that the global ranking approach provides the best performance only in light pollution scenario, but it is the local approach that performs better under heavy pollution. The heavy pollution attack causes more damage and has potential for intelligent attacks like whitewashing and collusion. So for most of our studies, we consider heavy pollution attacks and compare DRank with the local rating system.

6.3 Performance Analysis

We simulate DRank on the same platform and using the same default settings as we did in Chapter 5. The simulated network contains one server and 199 peers. By default,

each peer and the server is provided with $95KB/sec$ and $1MB/sec$ upload bandwidth, respectively, to support $64KB/sec$ streaming rate. The maximum upstream and downstream peers is set to 10 for each peer. The server can have at most 30 downstream peers. During the 10-minute streaming session, the peers join in the first 3 minutes following a Poisson distribution with $\lambda = 1.11$ and leave in the last 3 minutes following the same distribution. Among the peers, we select 10% peers as polluters to pollute aggressively. All peers maintain a 30-second initial delay period to ensure smooth playback.

6.3.1 Tuning parameters for DRank

Our study starts with tuning the local reputation management parameters. The reputation management at each peer in DRank is the same as the local reputation system, as described in Chapter 4. We vary different history and trust parameters to find the best setting. Table 6.3 and Table 6.4 show the NPI and playback quality, respectively. The playback skips and played polluted segments collectively provide the measurement of the playback quality. We place '—' in the table to specify that such setting produces similar results as the other settings do and '***' means that the setting is not suitable for smooth playback under the testing pollution model.

The parameters u and v determine how much reputation and trust history, respectively, are needed while updating those ratings. While updating the reputation and trust ratings of the neighbors, history helps defend against whitewashing and collusion attacks, respectively. As in Table 6.3 and Table 6.4, we find that under normal heavy attack, the best performance can be achieved with $u = 0.5$ and $v = 0.7$.

The parameter w determines how much second-hand information needs to be used. According to Table 6.3, the parameter w has a lower limit (0.15), above which the NPIs are very low. With lower values of w, a small amount of second-hand information is included, causing lack of information to calculate reputation ratings. It hinders the

Table 6.3: Impacts (NPI) of the history, second-hand information and the thresholds

	u	v	r	t	w
0	1.10	0.15	—	—	0.34
0.1	***	***	—	—	***
0.15	***	***	—	—	0.094
0.2	***	***	—	—	0.073
0.25	***	***	—	—	0.08
0.3	1.12	0.17	—	—	0.081
0.4	0.11	***	—	0.10	—
0.5	0.073	0.097	—	0.073	—
0.55	—	—	0.069	0.099	—
0.6	—	—	0.07	0.10	—
0.65	—	0.08	0.07	—	0.078
0.7	0.08	0.073	0.073	—	0.08
0.9	0.11	0.08	0.12	—	—

Table 6.4: Impacts (Playback quality) of the history, second-hand information and the thresholds

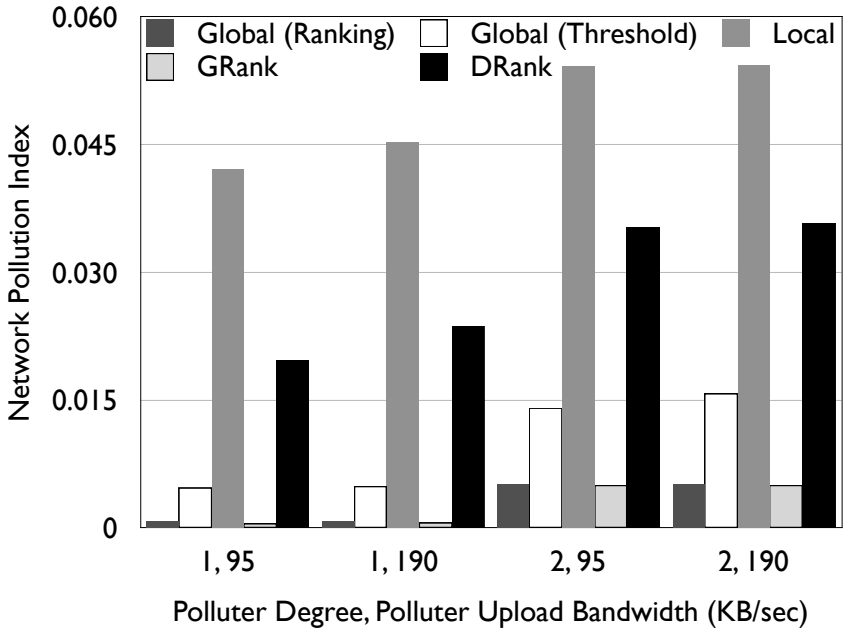
	Skip(%), Pollution(%)				
	u	v	r	t	w
0	33.49, 35.46	3.51, 13.44	***	***	2.29, 26.59
0.1	***	***	***	***	***
0.15	***	***	***	***	0.23, 9.23
0.2	***	***	***	***	0.11, 7.36
0.25	***	***	***	***	0.33, 7.81
0.3	3.09, 11.00	5.29, 14.67	***	***	0.51, 8.05
0.4	1.12, 9.30	***	***	0.48, 10.07	***
0.5	0.11, 7.36	1.96, 9.35	***	0.11, 7.36	***
0.55	—	***	4.04, 9.43	0.11, 9.64	***
0.6	—	***	1.63, 10.08	0.17, 9.24	***
0.65	—	0.59, 8.38	0.81, 10.18	—	0.78, 8.00
0.7	0.11, 8.02	0.11, 7.36	0.11, 7.36	—	1.49, 7.91
0.9	0.06, 10.37	0.63, 8.18	0.25, 11.71	—	***

polluter detection efficiency. We need to choose from the higher values of w. To better decide on the value of w, we look at the playback quality for different values. Higher values of w means that the neighbors' observations have a higher impact on a peer's decision. In the cases where the neighbors' observation is different from the first-hand information, a high value of w will cause a peer to disconnect from upstream peers more often. This causes a lack of available bandwidth in the neighborhoods leading to higher playback skips, as shown in Table 6.4. When combining the playback quality from Table 6.4 and NPI from Table 6.3, we choose the value of w as 0.2.

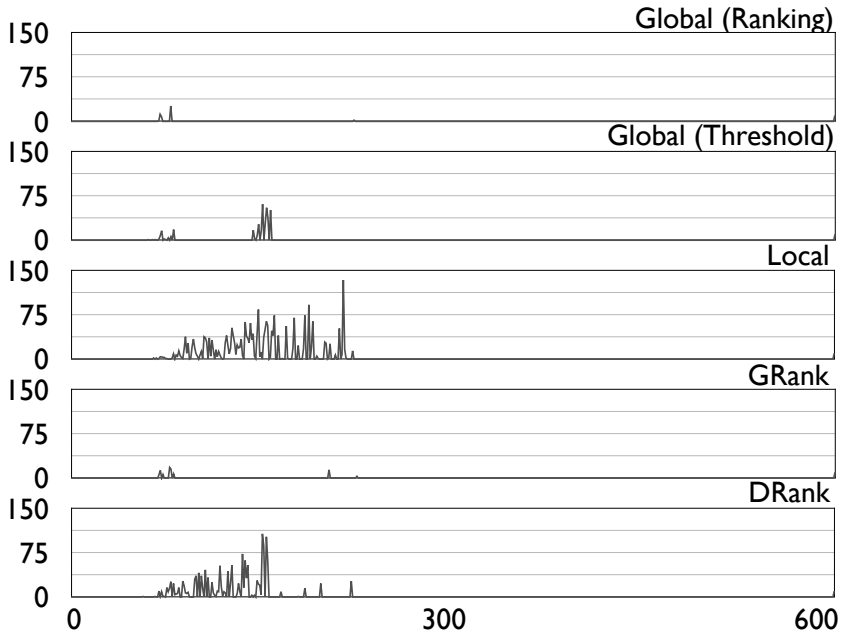
From Table 6.3 and Table 6.4, we also find the best values for r and t, which are the reputation and trust thresholds, respectively. While tuning the trust threshold t, Table 6.3 shows that the best NPI can be achieved with the value of 0.5. The playback quality in Table 6.4 also supports this conclusion. For the reputation threshold r, tighter threshold leads to a better performance, as shown in Table 6.3. According to the playback quality from Table 6.4, we determine the preferred value as 0.7.

6.3.2 Polluters' node degree and upload bandwidth

We confirmed in Chapter 5 that the node degree of the polluters is more influential than the upload bandwidth by experimenting with other defense techniques. Both GRank and DRank show the same trend as illustrated by Fig. 6.2(a). There are significant changes in NPI when we vary the polluters' node degree, but not much change when we vary polluters' upload bandwidth. Another important observation from Fig. 6.2 is that for all polluter formations, DRank, although showing better results than the local approach, does not perform better than the techniques that use the global approaches. This means that DRank cannot completely overcome the limitation of the local rating system due to the lack of global views. Since the global reputation ratings are saved centrally, more objective opinions about the polluters and the regular peers are compiled. When

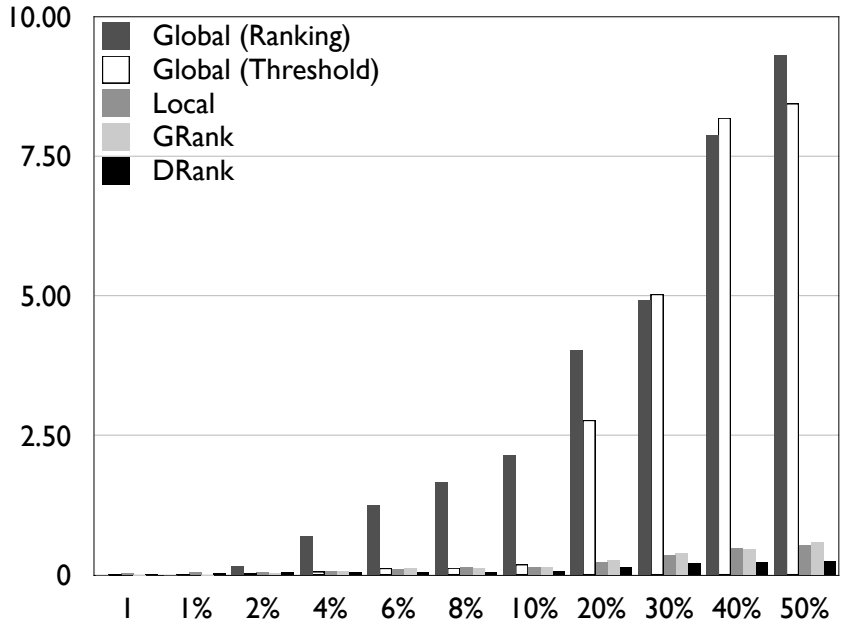


(a) NPI

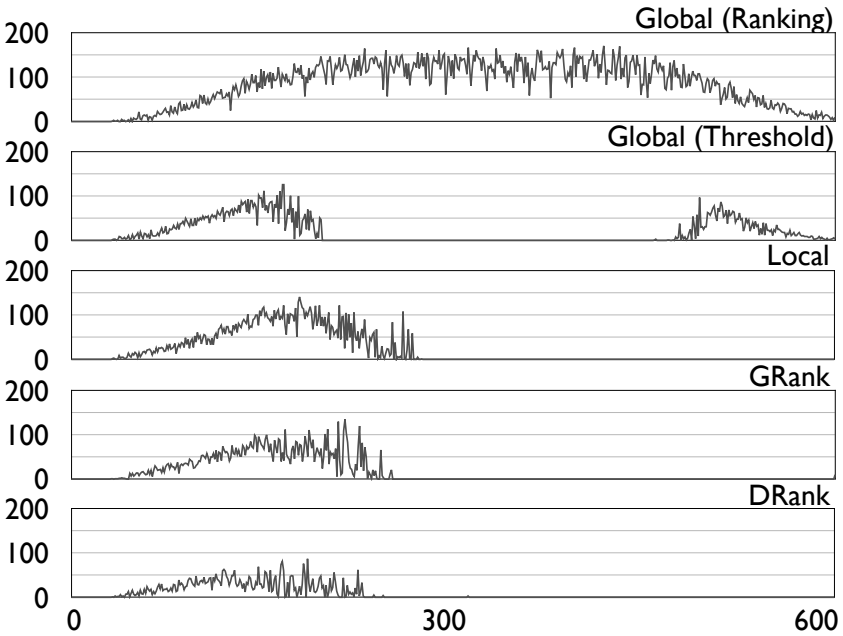


(b) # of polluted segments

Figure 6.2: The performance when varying the node degree and upload bandwidth, under a light pollution attack



(a) NPI



(b) # of polluted segments

Figure 6.3: Performance under increasing pollution attacks

the amount of polluters is very small, this central information can easily identify the polluters. Distributed approaches like local rating system and DRank takes more time to gather accurate information. Hence, it is not possible for the distributed approaches to be as accurate as the global approaches under light pollution attacks.

Under heavy attacks, however, distributed approaches become more robust than the global approaches, including GRank. Since the peers do not need to rely on the global ratings and can use their own judgement, they can remove a misbehaving peer as soon as they encounter enough polluted content from it. This helps the system to achieve lower NPI, as shown in Fig. 6.3(a). In DRank, the peers are gradually moving towards and forming good neighborhoods. Consequently, it is much more effective in determining and isolating the polluters and keeping the system clean afterwards. In addition, as shown in Fig. 6.3(b), while locating the polluters, the peers in DRank share less polluted content. Overall, DRank is the best defense technique in keeping the P2P streaming system clean.

According to Fig. 6.3, local and GRank perform almost the same under heavy attacks. Moreover, since we are interested in the fully distributed defense techniques for practical P2P streaming systems, for the rest of the experiments, we compare only DRank and the local rating system.

6.3.3 Polluter join time

Next, we focus on the flash crowd scenario where all polluters join the network at the same time. We vary the join time and compare the performance of DRank with the local defense technique, as shown in Fig. 6.4. As expected, DRank performs better than the local approach for all polluter join times. The NPI is not changed significantly but does exhibit a slight increase. It is mostly due to the ranking preference during the decision making. The later the polluters join, the more established the network is and the scarcer the bandwidth resource is. The later joined polluters will be selected by peers who are

thirsty for more upstream peers, which will give them more opportunity to pollute the network before being detected. When considering the playback quality in Table 6.5, the later joined polluters have more impact on the playback quality. However, there is no significant growth, and DRank outperforms the local defense technique in terms of NPI and the overall playback quality.

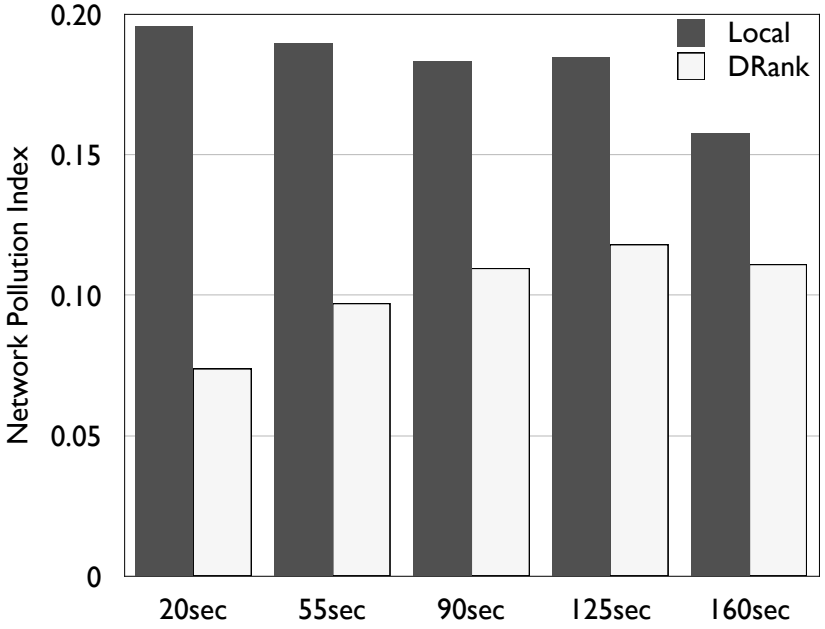


Figure 6.4: NPI for different polluter join time

Table 6.5: Playback quality for different polluter join times

	Skip(%), Pollution(%)	
Polluter Join Time	Local	DRank
20 sec	0.09, 16.50	0.80, 7.36
55 sec	0.11, 16.02	1.53, 9.41
90 sec	0.10, 15.42	2.81, 10.35
125 sec	0.08, 15.56	2.39, 11.10
160 sec	0.09, 13.44	0.42, 10.77

If we analize the playback quality, a trend can be observed in playback skips. The percentage of skips increases as the polluters join later and then starts to fall after 125sec.

This follows the same trend as shown in Fig. 6.4, for the same reason discussed earlier. As peers learn more about the polluters over time, they start to disconnect the polluters, which leads to a shortage of bandwidth in the neighborhood. Moreover, because of the ranking approach, the peers cancel polluters and try to connect to the small number of available good peers, which puts additional pressure on the upload bandwidth among the regular peers. So more skips are observed during this period. For the polluters who join at a much later time, the network is already established, and peers have more candidate upstream peers, *i.e.,* the polluters will have less chance to serve polluted content in this case. When a peer disconnects another peer for a better upstream peer, there will not be sudden loss of bandwidth.

6.3.4 Whitewashing attack

Next, we examine the performance of DRank under the whitewashing attack. We implement this attack as we did in Chapter 5. A polluter tries to improve its reputation by sending polluted segments following a certain probability P, instead of continuously sending them out. We vary P to observe whether the performance of DRank changes under this attack.

As shown in Fig. 6.5, DRank produces the worst NPI when $P = 0.2$. It then becomes more effective in keeping the NPI low as the pollution probability increases, where the local defense is less effective. The same conclusion holds for the playback quality, as reflected by Table 6.6. Especially, with low pollution probability, the amount of unusable segments (skips(%) + pollution(%)) becomes almost double than the highest probability. Since DRank balances the rating distribution by capping the α and β values, it can recover quickly from any false impression about the polluters. For this reason, the performance does not degrade beyond a certain level as opposed to the local ratings system.

Fig. 6.6 compares the change in reputation ratings of the polluters over time for

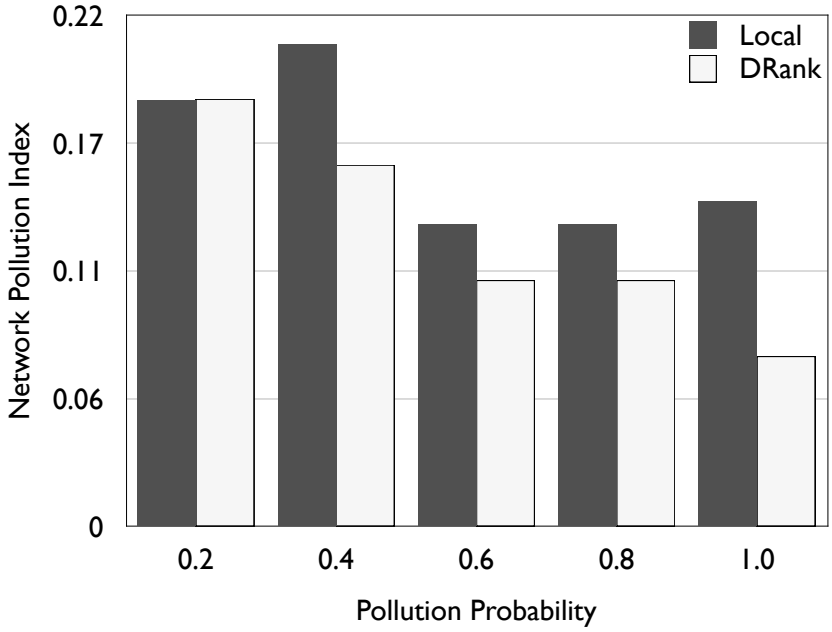
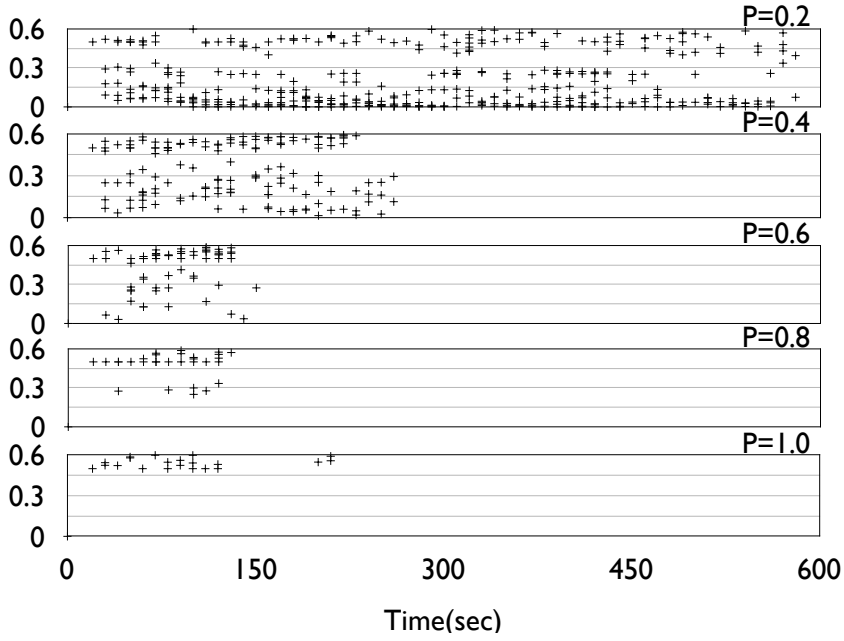


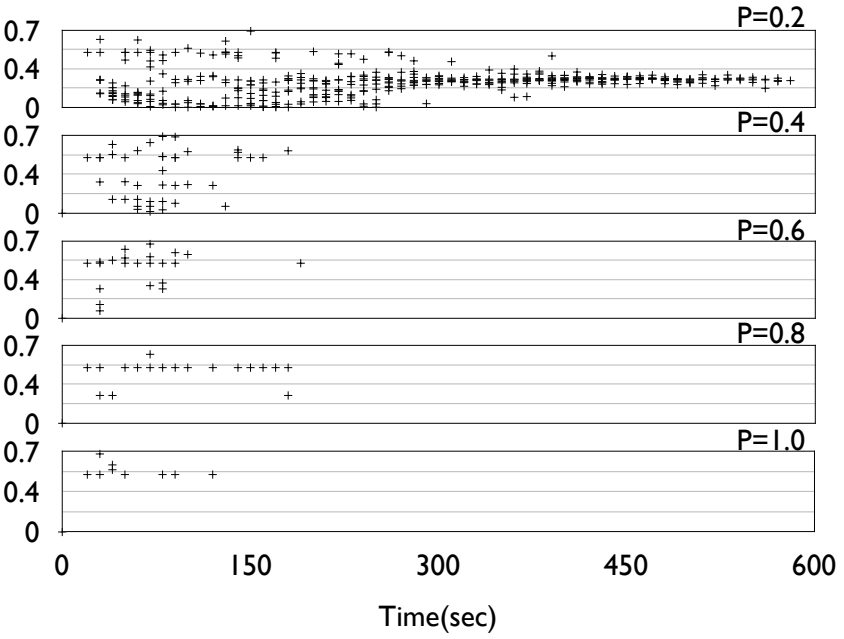
Figure 6.5: NPI for different pollution probabilities

Table 6.6: Playback quality for different pollution probabilities

	Skip(%), Pollution(%)	
Pollution Probability (P)	Local	DRank
0.2	0.15, 15.29	0.18, 15.43
0.4	0.28, 18.25	0.28, 13.97
0.6	0.10, 11.15	0.81, 10.55
0.8	0.14, 11.68	0.09, 10.42
1.0	0.15, 11.20	0.11, 7.36



(a) Local



(b) DRank

Figure 6.6: Change of polluter reputation with time

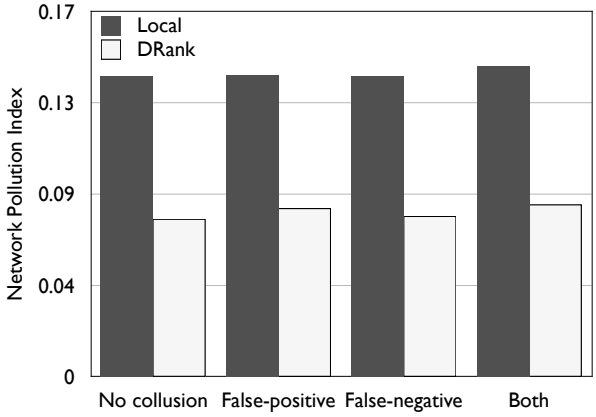
different pollution probabilities. With low pollution probability ($P = 0.2$), over time, the polluter reputation becomes concentrated around 0.25. This indicates that DRank keeps peers connected to polluters only if they have good reputations. Since the peers are being ranked locally, the only polluters having downstream peers are the ones with very good reputation ratings. These polluters pollute in such a way that this rating remains the same after it reaches the concentration point. For this reason, unlike the scattered ratings found in the local defense, the ratings in DRank converge. Hence, the polluters are hard to detect. This is why there is no improvement for DRank in this case. On the other extreme, for high pollution probability ($P = 1.0$), since there is no good reputable polluters as they are easier to be identified, very few polluters survive the rating and the ranking system.

6.3.5 Collusion attack

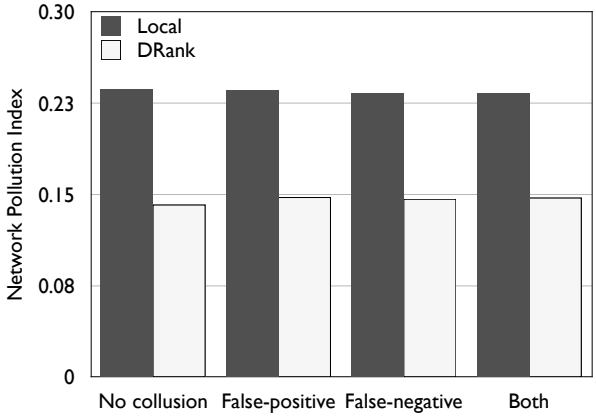
Finally, we turn our attention to the most challenging attack, the collusion attack. As in our experiments in Chapter 5, our polluters act as colluders. They launch the attack by circulating good ratings about other polluters, or circulating bad ratings about regular peers, or both. We examine DRank under these three scenarios.

The local reputation broadcast employed by the local approach and DRank creates opportunities for the collusion attacks. However, the trust ratings do not allow the colluders to improve their reputation by disseminating false information. For this reason, according to Fig. 6.7, there is no significant change in NPI for both the local defense and DRank under different collusion scenarios. We find that no matter how many colluders participate in the collusion attack, both the local defense and DRank can detect it and keep the NPI under control. In general, the performance of DRank for all collusion scenarios is significantly better than the local approach.

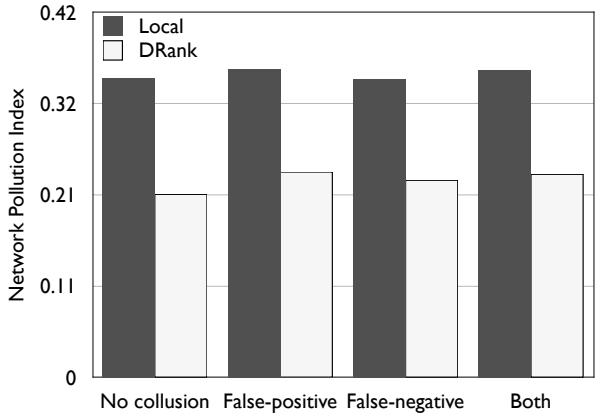
The collusion attack can have impacts on the parameters controlling the second-hand



(a) 10% polluter



(b) 20% polluter



(c) 30% polluter

Figure 6.7: Effect of collusion attack

Table 6.7: Performance (NPI) for different DRank parameters

Parameter values	v	t	w
0.0	0.08	***	0.34
0.1	—	***	0.12
0.2	—	***	0.08
0.3	0.09	0.27	0.08
0.4	—	0.11	0.09
0.5	0.08	0.08	—
0.6	—	0.09	—
0.7	0.08	—	—

Table 6.8: Performance (playback quality) for different DRank parameters

	Skip(%), Pollution(%)		
Parameter values	v	t	w
0.0	1.56, 7.64	***	3.01, 26.49
0.1	***	***	2.11, 13.50
0.2	***	***	0.86, 7.92
0.3	2.00, 9.28	9.99, 20.95	0.9, 8.78
0.4	***	0.38, 10.67	1.81, 9.0
0.5	1.08, 8.34	0.86, 7.92	***
0.6	***	0.81, 9.73	***
0.7	0.86, 7.92	***	***

information. In Table 6.7 and Table 6.8, we vary these parameters again to see the collusion effect. For the NPI from Table 6.7, collusion does not change the effect of the trust history parameter v. However, the playback quality in Table 6.8 indicates that it is better to keep the trust history. The lower values of v allow conceiving more illegitimate segments than the higher values do. Since colluders provide a false impression, the peers need to rely more on history to avoid being fooled.

Tighter (smaller) trust rating (t) thresholds cause less second-hand information to be considered, and loose (bigger) thresholds cause more false information to be considered. The impact of different choices for t is significant since only a small fraction of second-hand information is taken into the reputation ratings. However, less second-hand information makes peers less knowledgeable about the network, which allows a polluter to pollute more just by changing its neighborhood. Therefore, we find that the sweet spot for the t value is 0.5 according to Table 6.7 and Table 6.8.

Similarly, there is also a sweet spot among the different choices of w (the weight factor for the second-hand information). Too much and too little second-hand information both will make the system more vulnerable to collusion attacks. Too much second-hand information implies that the peers can be easily fooled by a polluter. Too little second-hand information makes the peers unaware of the network status. According to the NPI in Table 6.7 and the playback quality in Table 6.8, we find that the best value for w is 0.2.

Compared to the results from Sec. 5.8, DRank is much more flexible in handling the collusion attacks. It outperforms the local defense regardless of the setting for v, w and t.

6.4 Summary

In this chapter, we explored two possible approaches of combining the local and global reputation systems and performed a comparison study with the existing global and local reputation systems. From the experiments in this chapter, we find that although DRank is still not the best defense against light pollution, it is the best for heavy attacks, and is better than the other distributed reputation-based approaches, namely the local rating system and GRank for all amount of pollution. In fact, under heavy attack, DRank performs the best in terms of both the NPI and the playback quality. In addition, DRank is less sensitive to network settings and parameter tuning. Finally, it is a robust solution for P2P streaming systems to fight against pollution attacks for three reasons: First, it is fully distributed, which helps it easily adopt with the network dynamics of P2P network and avoid targeted attacks. Second, it is a dynamic solution, as it encourages peers to move to better neighborhoods based on their perception about the neighborhoods instead of the entire network. Third, it takes less time to detect polluters - even less than the local rating system.

We not only proposed the new distributed rank-based reputation rating approach, we compared all five reputation-based approaches. We found that, being a fully distributed approach, DRank is the most flexible and effective defense mechanism to prevent pollution attacks in P2P streaming systems.

Chapter 7

Conclusion

This thesis presents a study of reputation-based defenses against pollution attack, one of the most devastating attacks against P2P networks, in its various forms. Pollution attacks not only degrade the content quality, but also can be used to spread viruses, malware, and bots. This thesis performs a thorough investigation of the defense techniques by testing them under various challenges, thus providing insight into their effectiveness. In this chapter, we present a summary of contributions and thoughts for future work.

7.1 Contributions

We studied the performance of the defense techniques under critical conditions. First, we simulated them on P2P live streaming system, where the biggest challenge is the time restrictions on segment collection, thus, maintaining the playback quality. Second, we not only tested them against simple pollution attack, but also combined with it attacks like flash crowd, whitewashing, and collusion. The contributions are two-fold: 1) We measured the strength of some existing defense techniques to find out the key requirements of a robust defense mechanism, 2) With our findings, we proposed a defense mechanism with better performance. Our experimental results show that centralized global knowledge of the content flow in the network does not necessarily improve performance. It is often susceptible to collaborative attacks. We also find that, to prevent attacks, expelling misbehaving peers is often more useful than limiting their likelihood to be connected, although this can lead to poor playback quality. Along with these two key findings, conclusions drawn from our extensive study are summarized here.

- The node degree of the polluters is a more determining factor than the upload bandwidth is. In other words, increasing the number of downstream peers of the polluters will significantly increase the amount of polluted content in the network.
- The time when the polluters join the system has little impact on the severity of the attack.
- For systems under light pollution attacks, the global ranking system is the best defense technique as it can identify the polluter without expelling any peer from the system. However, for systems under heavy pollution attacks, the threshold-based defense mechanisms are more effective as it is important to cut off the polluters, even with the cost of expelling innocent peers and degrading the playback quality.
- The global approaches are susceptible to collusion attacks because they do not have any way to verify the trustworthiness of peers. The global normalization may give the polluters higher rating, especially when the network size is small.
- The local rating system yields steady performance not only against heavy pollution attacks, but also against collaborative attacks because of its trust verification consideration. However, it is not effective in fighting against light pollution attacks.
- Rate calculation using lengthy history of the first-hand and second-hand information is more accurate, but keeping too much history can make it difficult to omit reports from the polluters that are performing a collusion attack.

Our proposed approach uses the ranking concept from the global ranking approach and combines it with the local rating system. We explored two possibilities, namely GRank and DRank. GRank, using raw combination, performs as good as the global ranking with light pollution, but cannot perform as good as the local rating system does.

Moreover, this approach still uses centralization. DRank distributes the ranking at peer-level to make the system fully distributed. We find that, although DRank cannot achieve the performance of GRank in the light pollution scenario, it performs the best under heavy attacks. Analyzing the new approach, we have the following findings:

- The fully distributed approaches cannot decide on the polluters fast enough, since it does not have centrally stored global information about the peers. As a result, under light attacks, the polluter detection takes time.
- It is wise to gradually situate peers in good neighborhoods instead of doing it immediately. Gradual movement of peers to good neighborhoods distributes the polluters in different neighborhoods. So expelling them will not cause sudden loss of bandwidth in a certain neighborhood.
- Normalization of reputation ratings is not always a good idea to compare among peers, since innocent peers may become victim. Rather, if the peers are treated separately, the performance can improve.

7.2 Future Works

We simulated the reputation-based approaches on a live-streaming system. Live streaming systems have time restriction challenge, which is a good test-bed for testing defense compatibilities. However, other P2P systems like P2P VoD and P2P file-sharing systems have their own challenges. For example, in VoD, there are several sessions as one live streaming session and in P2P file sharing system, the polluters have more time to pollute segments, which helps it pollute more intelligently. Performance measurement studies can be conducted in these systems to examine the effectiveness of the defense techniques. It will be interesting to see how DRank performs when applied in these systems.

DRank, although performs better under heavy pollution, is still challenged by the light pollution. In fact, as we mentioned earlier, the distributed approaches are susceptible to light pollution attacks, compared to the global approaches. To cope with the dynamics of P2P networks, we need a fully distributed solution, which the global reputation approaches do not promise. Research can be launched to explore this specific area to come up with a solution that can perform better under light pollution without hampering the performance under heavy attacks.

Bibliography

[1] *http://www.youtube.com/.*

[2] *http://www.bbc.co.uk/iplayer/.*

[3] *http://www.pplive.com/en/index.html*, PPLive. [Online]. Available: http://www.pplive.com/en/index.html

[4] *http://www.uusee.com/*, UUSee. [Online]. Available: http://www.uusee.com/

[5] J. Liang, R. Kumar, Y. Xi, and K. W. Ross, "Pollution in P2P File Sharing Systems," in *Proc. of the 24th IEEE Computer and Communications Societies (INFOCOM)*, vol. 2, Miami, FL, March 13-17 2005, pp. 1174–1185.

[6] P. Dhungel, X. Hei, K. W. Ross, and N. Saxena, "Pollution in P2P Live Video Streaming," *International Journal of Computer Networks & Communications (IJCNC)*, vol. 1, no. 2, pp. 99–110, July 2009.

[7] N. Christin, A. S. Weigend, and J. Chuang, "Content Availability, Pollution and Poisoning in File Sharing Peer-to-Peer Networks," in *Proc. of 6th ACM Conference on Electronic Commerce (EC)*, Vancouver, Canada, June 5-8 2005, pp. 68–77.

[8] R. Kumar, D. D. Yao, A. Bagchi, K. W. Ross, and D. Rubenstein, "Fluid Modeling of Pollution Proliferation in P2P Networks," in *Proc. of ACM Joint International Conference on Measurement and Modeling of Computer Systems, SIGMETRICS/Performance*, Saint Malo, France, June 26-30 2006, pp. 335–346.

[9] P. Dhungel, X. Hei, K. W. Ross, and N. Saxena, "The Pollution Attack in P2P Live Video Streaming: Measurement Results and Defenses," in *Proc. of ACM Workshop on Peer-to-peer Streaming and IP-TV (P2P-TV 2007)*, Kyoto, Japan, August 27-31 2007, pp. 323–328.

[10] E. Lin, D. M. N. de Castro, M. Wang, and J. Aycock, "SPoIM: A Close Look at Pollution Attacks in P2P Live Streaming," in *Proc. of the 18th International Workshop on Quality of Service (IWQoS)*, Beijing, China, June 16-18 2010, pp. 1–9.

[11] B. Hu and H. V. Zhao, "Joint Pollution Detection and Attacker Identification in Peer-to-Peer Live Streaming," in *Proc. of IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, Dallas, Texas, March 14-19 2010, pp. 2318–2321.

[12] D. S. Wallach, "A Survey of Peer-to-Peer Security Issues," in *Proc. of Mext-NSF-JSPS International Conference on Software Security: Theories and Systems (ISSS)*, Tokyo, Japan, November 8-10 2002, pp. 42–57.

[13] J. Liang, N. Naoumov, and K. W. Ross, "Efficient Blacklisting and Pollution-Level Estimation in P2P File-Sharing Systems," in *Proc. of the 1st Asian Internet Engineering conference on Technologies for Advanced Heterogeneous Networks (AINTEC)*, Bangkok, Thailand, December 13-15 2005, pp. 1–21.

[14] A. B. Vieira, S. Campos, and J. Almeida, "Fighting Attacks in P2P Live Streaming. Simpler is Better," in *Proc. of IEEE international conference on Computer Communications Workshops (ICCCW), in conjunction with INFOCOM*, Rio de Janeiro, Brazil, April 19-25 2009, pp. 355–356.

[15] J. A. A. Borges and S. Campos, "Fighting Pollution in P2P Live Streaming Systems," in *Proc. of IEEE International Conference on Multimedia & Expo (ICME)*, Hannover, Germany, June 23-26 2008, pp. 481–484.

[16] C. Costa and J. Almeida, "Reputation Systems for Fighting Pollution in Peer-to-Peer File Sharing Systems," in *Proc. of 7th IEEE International Conference on Peer-to-*

Peer Computing, Galway, Ireland, September 2-5 2007, pp. 53–60.

[17] S. Buchegger and J. Y. L. Boudec, "A Robust Reputation System for P2P and Mobile Ad-hoc Networks," in *Proc. of the 2nd Workshop on the Economics of Peer-to-Peer Systems (P2PEcon)*, Harvard University, June 4-5 2004, pp. 1–6.

[18] T. G. Papaioannou and G. D. Stamoulis, "Effective Use of Reputation in Peer-to-Peer Environments," in *Proc. of IEEE International Symposium on Cluster Computing and the Grid*, Chicago, IL, April 19-22 2004, pp. 259–268.

[19] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara, "Reputation Systems," *Communications of the ACM*, vol. 43, no. 12, pp. 45–48, December 2000.

[20] C. K. Wong and S. S. Lam, "Digital Signatures for Flows and Multicasts," *IEEE/ACM Transactions on Networking*, vol. 7, no. 4, pp. 504–513, August 1999.

[21] K. J. R. L. H. Vicky Zhao, "Fingerprint Multicast in Secure Video Streaming," in *IEEE Transactions on Image Processing*, 2006.

[22] A. Habib, D. Xu, M. Atallah, B. Bhargava, and J. Chuang, "Verifying Data Integrity in Peer-to-Peer Media Streaming," in *Proc. of the 12th Annual Multimedia Computing and Networking (MMCN)*, San Jose, CA, January 16-20 2005, pp. 1–12.

[23] L. Xiong and L. Liu, "PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 7, pp. 843–857, July 2004.

[24] K. Aberer and Z. Despotovic, "Managing Trust in a Peer-to-Peer Information System," in *Proc. of the 9th International Conference on Information and Knowledge Management (CIKM)*, McLean, VA, November 6-11 2001, pp. 310–317.

[25] F. Cornelli, E. Damiani, D. C. di Vimercati, S. Paraboschi, and P. Samarati, "Choosing Reputable Servents in a P2P Network," in *Proc. of the 11th ACM International Conference on World Wide Web (WWW)*, Honolulu, Hawaii, May 7-11 2002, pp. 376–386.

[26] E. Damiani, D. C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks," in *Proc. of the 9th ACM Conference on Computer and Communications Security*, Washington D.C., November 17-21 2002, pp. 207–216.

[27] A. A. Selcuk, E. Uzun, and M. R. Pariente, "A Reputation-Based Trust Management System for P2P Networks," in *Proc. of IEEE International Symposium on Cluster Computing and the Grid*, Chicago, IL, April 19-22 2004, pp. 251–258.

[28] L. Mekouar, Y. Iraqi, and R. Boutaba, "Detecting Malicious Peers in a Reputation-Based Peer-to-Peer System," in *Proc. of Consumer Communications & Networking Conference (CCNC)*, Las Vegas, Nevada, January 3-6 2005, pp. 1–6.

[29] R. Zhou and K. Hwang, "Gossip-Based Reputation Aggregation for Unstructured Peer-to-Peer Networks," in *Proc. of IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Long Beach, CA, March 26-30 2007, pp. 1–10.

[30] A. Singh and L. Liu, "TrustMe: Anonymous Management of Trust Relationships in Decentrlized P2P Systems," in *Proc. of 3rd IEEE International Conference on Peer-to-Peer Computing*, Linkoping, Sweden, September 1-3 2003, pp. 142–149.

[31] D. Dutta, A. Goel, R. Govindan, and H. Zhang, "The Design of a Distributed Rating Scheme for Peer-to-Peer Systems," in *Proc. of the 1st Workshop on Economics of Peer-to-Peer Systems (P2PEcon)*, Berkeley, CA, June 5-6 2003, pp. 1–5.

[32] R. Zhou and K. Hwang, "PowerTrust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, pp. 460–273, April 2007.

[33] A. Singh, T.-W. Ngan, P. Druschel, and D. S. Wallach, "Eclipse Attacks on Overlay Networks: Threats and Defenses," in *Proc. of the 25th IEEE Computer and Communications Societies (INFOCOM)*, Barcelona, Spain, April 23-29 2006, pp. 1–12.

[34] H. Johansen, A. Allavena, and R. van Renesse, "Fireflies: Scalable Support for Intrusion-Tolerant Network Overlays," in *Proc. of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems (EuroSys)*, Leuven, Belgium, April 18-21 2006, pp. 3–13.

[35] M. Haridasan and R. van Renesse, "Defense Against Intrusion in a Live Streaming Multicast System," in *Proc. of the 6th IEEE International Conference on Peer-to-Peer Computing (P2P)*, Cambridge, UK, September 6-8 2006, pp. 185–192.

[36] M. Haridasan, I. Jansch-Porto, and R. van Renesse, "Enforcing Fairness in a Live-Streaming System," in *Proc. of Multimedia Computing and Networking (MMCN)*, San Jose, CA, January 30-31 2008, pp. 1–12.

[37] S. Shetty, P. Galdames, W. Tavanapong, and Y. Cai, "Detecting Malicious Peers in Overlay Multicast Streaming," in *Proc. of 31st IEEE Conference on Local Computer Networks (LCN)*, Tampa, FL, November 14-16 2006, pp. 499–506.

[38] D. Grolimund, L. Meisser, S. Schmid, and R. Wattenhofer, "Havelaar: A Robust and Efficient Reputation System for Active Peer-to-Peer Systems," in *Proc. of the 1st Workshop on the Economics of Networked Systems (NetEcon)*, Ann Arbor, Michigan, June 11 2006, pp. 69–74.

[39] S. Yang, H. Jin, B. Li, and X. Liao, "A Modeling Framework of Content Pollution in Peer-to-Peer Video Streaming Systems," *Elsevier Computer Networks*, vol. 53, pp. 2703–2715, June 2009.

[40] G.-G. Chun, B. Y. Zhao, and J. D. Kubiatowicz, "Impact of Neighbour Selection on Performance and Resilience of Structured P2P Networks," in *Proc. of the 4th International Workshop on Peer-to-Peer Systems (IPTPS)*, Ithaca, NY, February 24-25 2005, pp. 264–274.

[41] R. Jurca and B. Faltings, "An Incentive Compatible Reputation-Mechanism," in *Proc. of the 2nd International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS)*, Melbourne, Australia, July 14-18 2003, pp. 1026–1027.

[42] Incentives for Cooperation in Peer-to-Peer Networks, "K. lai and m. feldman and i. stocia and j. chuang," in *Proc. of the 1st Workshop on Economics of Peer-to-Peer Systems (P2PEcon)*, Berkeley, CA, June 5-6 2003, pp. 1–6.

[43] To Share or Not to Share: An Analysis of Incentives to Contribute in Collaborative File Sharing Environments, "K. ranganathan and m. ripeanu and a. sarin and i. foster," in *Proc. of the 1st Workshop on Economics of Peer-to-Peer Systems (P2PEcon)*, Berkeley, CA, June 5-6 2003, pp. 1–6.

[44] F. Benevenuto, C. Costa, M. Vasconcelos, V. Almeida, J. Almeida, and M. Mowbray, "Impact of Peer Incentives on the Dissemination of Polluted Content," in *Proc. of the 21st Annual ACM Symposium on Applied Computing (SAC)*, Dijon, France, April 23-27 2006, pp. 1875–1879.

[45] eBay Inc., *http://www.ebay.com*.

[46] *http://www.amazon.com*, Amazon.com, Inc.

[47] C. Dellarocas, "Analyzing the Economic Efficiency of eBay-like Online Reputation Reporting Mechanisms," in *Proc. of the 3rd ACM Conference on Electronic Commerce (EC)*, Tampa, FL, October 14-17 2001, pp. 171–179.

[48] ——, "Immunizing Online Reputation Reporting Systems Against Unfair Ratings and Discriminatory," in *Proc. of the 2nd ACM Conference on Electronic Commerce (EC)*, Minneapolis, MN, October 17-20 2000, pp. 150–157.

[49] Q. Lian, Z. Zhang, M. Yang, B. Y. Zhao, Y. Dai, and X. Li, "An Empirical Study of Collusion Behavior in the Maze P2P File-Sharing System," in *Proc. of the 27th IEEE International Conference on Distributed Computing Systems (ICDCS)*, Toronto, Canada, June 25-29 2007, pp. 56–65.

[50] W. Conner, K. Nahrstedt, and I. Gupta, "Preventing DoS Attacks in Peer-to-Peer Media Streaming Systems," in *Proc. of the 13th Annual Multimedia Computing and Networking (MMCN) Proc. of the 12th Annual Multimedia Computing and Networking (MMCN) Proc. of the 12th Annual Multimedia Computing and Networking (MMCN) Proc. of the 13th Annual Multimedia Computing and Networking (MMCN)*, San Jose, CA, January 15-18 2006, pp. 1–12.

[51] A. Singh, M. Castro, P. Druschel, and A. Rowstron, "Defending Against Eclipse Attacks on Overlay Networks," in *Proc. of the 11th ACM SIGOPS European Workshop (EW)*, Leuven, Belgium, September 19-22 2004, pp. 1–6.

[52] W. Wang, Y. Xiong, Q. Zhang, and S. Jamin, "Ripple-Stream: Safeguarding P2P Streaming Against DoS Attacks," in *Proc. of IEEE International Conference on Multimedia & Expo (ICME)*, Toronto, Canada, July 9-12 2006, pp. 1417–1420.

[53] X. Jin, S.-H. G. Chan, W.-P. K. Yiu, Y. Xiong, and Q. Zhang, "Detecting Malicious Hosts in the Presence of Lying Hosts in Peer-to-Peer Streaming," in *Proc. of IEEE*

International Conference on Multimedia & Expo (ICME), Toronto, Canada, July 9-12 2006, pp. 1537–1540.

[54] K. Walsh and E. G. Sirer, "Fighting Peer-to-Peer SPAM and Decoys with Object Reputation," in *Proc. of the 3rd Workshop on the Economics of Peer-to-Peer Systems (P2PEcon)*, Philadelphia, PA, August 22 2005, pp. 138–143.

[55] ——, "Experience with an Object Reputation System for Peer-to-Peer Filesharing," in *Proc. of the 3rd USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, San Jose, CA, May 8-10 2006, pp. 1–14.

[56] S. Marti and H. Garcia-Molina, "Identity Crisis: Anonymity v.s. Reputation in P2P Systems," in *Proc. of 3rd IEEE International Conference on Peer-to-Peer Computing*, Linkoping, Sweden, September 1-3 2003, pp. 134–141.

[57] R. Dingledine, N. Mathewson, and P. Syverson, "Reputation in P2P Anonymity Systems," in *Proc. of the 1st Workshop on Economics of Peer-to-Peer Systems (P2PEcon)*, Berkeley, CA, June 5-6 2003, pp. 1–6.

[58] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "CoolStreaming/DONet: A Data-Driven Overlay Network for Peer-to-Peer Live Media Streaming," in *Proc. of the 24th Conference of the IEEE Communications Society (INFOCOM 2005)*, vol. 3, Miami, FL, March 13-17 2005, pp. 2102–2111.

[59] J. Seibert, X. Sun, C. Nita-Rotaru, and S. Rao, "Towards Securing Data Delivery in Peer-to-Peer Streaming," in *Proc. of the 2nd International Conference on Communication Systems and Networks*, 2010.

[60] B. Hu and H. V. Zhao, "Pollution-resistant peer-to-peer live streaming using trust management," in *16th IEEE International Conference on Image Processing (ICIP)*, 2009.

[61] Y. Li and J. C. S. Lui, "Stochastic Analysis of a Randomized Detection Algorithm for Pollution Attack in P2P Live Streaming Systems," *Elsevier Performance Evaluation*, vol. 67, pp. 1273–1288, August 2010.

[62] S. Yang, H. Jin, B. Li, X. Liao, H. Yao, and X. Tu, "The Content Pollution in Peer-to-Peer Live Streaming Systems: Analysis and Implications," in *Proc. of the 37th International Conference on Parallel Processing (ICPP)*, Portland, Oregon, September 9-12 2008, pp. 652–659.