Impact of Peer Incentives on the Dissemination of Polluted Content

Fabricio Benevenuto
UFMG, Belo Horizonte/Brazil
fabricio@dcc.ufmg.br

Cristiano Costa
UFMG, Belo Horizonte/Brazil
krusty@dcc.ufmg.br

Marisa Vasconcelos
UFMG, Belo Horizonte/Brazil
isa@dcc.ufmg.br

Virgilio Almeida
UFMG, Belo Horizonte/Brazil
virgilio@dcc.ufmg.br

Jussara Almeida
UFMG, Belo Horizonte/Brazil
jussara@dcc.ufmg.br

Miranda Mowbray
HP Labs, Bristol/UK
miranda.mowbray@hp.com

ABSTRACT

Recent studies have reported a new form of malicious behavior in file-sharing Peer-to-Peer systems: content pollution. The dissemination of polluted content in a P2P system has the detrimental effect of reducing content availability, and ultimately, decreasing the confidence of users in such systems. Two potential strategies for polluting P2P content are decoy insertion, which consists of injecting corrupted copies of a file into the system, and hash corruption, which consists of injecting a corrupted file with the same hash code as a non-corrupted one. Polluted content disseminates through P2P networks because users typically do not delete the corrupted files that they download.

This paper investigates the effectiveness of peer incentives to delete corrupted files in reducing the dissemination of polluted content, considering the two aforementioned pollution mechanisms. Our simulation results show that the effectiveness of incentives is highly dependent on the pollution mechanism. We show that for a pollution dissemintation techinique called hash corruption, only effective incentive mechanisms are able to avoid spreading of polluted content.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems

General Terms

Security

Keywords

Peer-to-Peer Networks, Content Pollution, Peer Incentive

SAC'06 April 23-27, 2006, Dijon, France

1. INTRODUCTION

The growth of the number of files shared, the number of users, and the amount of Internet traffic associated with Peer-to-Peer (P2P) systems have been widely discussed recently. In fact, P2P has experienced a dramatic growth since its inception, and now contributes to a large proportion of all Internet traffic [16]. P2P systems have also experienced an important growth in complexity. They have evolved from simple collections of peers connected to a central server, such as Napster [10], to much more complex distributed systems, hierarchically organized, with several features such as swarm downloading and reputation mechanisms [1].

The quick evolution of P2P systems has had impact not only on Internet traffic, but also, importantly, on the music and video recording industries. These industries have experienced, as peer-to-peer systems have spread, losses of millions of dollars in CD and DVD sales. Since the legal action against Napster [8], we have seen the music and video industries determined to start a war against piracy and consequently against the peer-to-peer file-sharing systems that enable it. With the emergence of decentralized P2P systems such as Gnutella [6], KaZaA [7], eDonkey [4], and BitTorrent [2], the music industry became unable to prosecute a P2P system itself, since these decentralized systems do not have a central server. After this development, the music industry tried to sue individual users of P2P applications for copyright infringements, without much success.

However, a recent study [9] has shown some evidence that intervention by the music industry involving the dissemination of polluted content on P2P networks has been relatively successful. Pollution consists of spreading copies of a specific file (a particular song or movie), with the same metadata (i.e. name, artist), but with corrupted content [3]. By using this mechanism, a music company can decrease the popularity of the file, by making it more difficult for users to download a correct copy, and can reduce the popularity of the system as a whole. For instance, J. Liang et al. [9] showed that more than 50% copies and versions of a popular file that were found by searching the FastTrack network [5] were polluted.

There are several mechanisms for disseminating polluted content. We will call the most well-known one *decoy insertion*. When a user searches for a file, the P2P system returns versions of that file, where each version has a number

of copies. *Decoy insertion* consists of the insertion of bogus versions of a file into the network, in order to decrease the probability that a user will find a non-polluted copy. Another important mechanism for disseminating polluted content is to insert a corrupted file into the P2P network which has the same hash code as a non-polluted file. We call this mechanism *hash corruption*.

Peer-to-peer systems rely on cooperation among users to increase the variety of services provided by this type of system. We can assume that incentive techniques can be adopted by users, based on the assumption that cooperation will emerge because users want to download non-polluted files [13]. Thus, we want to understand the effect of the use of incentives to remove polluted files from a peer-to-peer system. A possible way of reducing pollution is to give users incentives to identify corrupted files. This identification could be done by ranking files for quality or deleting polluted files. In [9] it was shown that only a few users rank the files they download, and we believe that this explains the high level of pollution in some systems.

In this context, we have evaluated the effectiveness of such incentives on the dissemination of polluted content, considering the two pollution mechanisms described above. However we are not proposing specific incentives or reputation mechanisms; we are interested in the effectiveness of incentives, rather than their design.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 provides an overview of the two mechanisms of pollution dissemination in P2P networks. Section 4 presents our evaluation methodology, describing our P2P simulator and the metrics we used in the performance evaluation. The main results are presented in Section 5. Section 6 offers conclusions and directions for future work.

2. RELATED WORK

There are only a few studies on content pollution in P2P networks. Pollution in the FastTrack network [5] as mentioned above, was analyzed in [9]. The authors developed a crawler able to search and download files in this network. They showed that pollution is pervasive in this P2P network and discussed several anti-pollution mechanisms. Liang et al. [3] distinguish between two kinds of pollution: they use the term poisoning to refer to intentional corruption of P2P content, and reserve the term pollution to refer to accidental injection of corrupted content into the network. They evaluated content availability and content pollution in three major existing P2P systems: KaZaA [7], eDonkey [4] and Gnutella [6]. Pouwelse et al. [12] analyzed the integrity of BitTorrent/Supernova by trying to insert polluted content into the system, and showed that, in this system, the efficiency of moderators made content corruption more difficult. However, this efficiency is achieved through the centralized nature of the standard BitTorrent search engine, which becomes a possible single point of failure.

These studies provided the first analysis of content pollution in P2P networks, and showed that pollution really can be pervasive in such systems. However, they did not evaluate how polluted content is disseminated, and did not explore the techniques deployed by music companies for spreading pollution. Furthermore, they only considered decoy insertion as the mechanism used. Although there is only anecdotal evidence of content pollution caused by hash corruption, several discussions of pollution caused this way can be found on the Internet. For instance, a company called Viralg claims to be able to eradicate chosen content from a P2P network by using hash corruption [18].

Other threats to P2P systems are spyware and worms. There have been some recent studies on the spread of worms and spyware among P2P users. The authors of [15] measured the spread of spyware using traces from a University environment. In [20] there is a proposal and validation of an analytical model for evaluating the propagation of worm attacks in a P2P network. However, there is a significant difference between pollution dissemination and worm dissemination. The propagation of polluted content is carried out by user requests, whereas worms may propagate by itself.

3. MECHANISMS FOR DISSEMINATING POLLUTED CONTENT

This Section describes two well-known mechanisms for disseminating polluted content: the decoy insertion mechanism and the hash corruption mechanism. We evaluated the effect of incentives when corrupt content is spread using these two techniques.

3.1 Decoy Insertion

Decoy insertion is a common sabotage mechanism used in P2P networks in which corrupted versions of a particular file are inserted into the network in order to make it difficult for users to find an uncorrupted version of that file. The corrupted file that is inserted, which we call a decoy, contains the same metadata as the polluted file. Usually, when a user searches for a file, the P2P application groups the available copies into different versions, and presents the versions with the largest numbers of copies to the user. If users do not delete polluted files as soon as they are downloaded, the decoys inserted into the system may be copied many times, making it difficult to find non-polluted content.

There are companies such as Viralg [18], RetSpan [14] and OverPeer [11] which offer services to protect any content from non-authorized distribution on P2P networks. They use a large number of machines sharing a large number of polluted versions of the target file, each version having many copies.

3.2 Hash Corruption Mechanism

In a P2P system, when a user starts sharing a file, a unique ID is associated with the file. This ID allows applications to identify the files that the users share. Moreover, when a user receives the result of a search, the P2P client groups results with the same ID, so that a file can be downloaded from multiple sources simultaneously. This ID is generated by applying a hash function to the file content, and each system uses a different algorithm to create it.

P2P systems assume that a file ID generated using the hash function is unique. However, it is possible to have two different files with the same ID. Some of the common used algorithms generate the file ID based only on parts of the file. In this context, a malicious peer can make changes on the parts of the file which are not used by the algorithm to generate the file ID, creating different files with different file IDs. When a user requests this file, it will receive a list of versions of that file, each one with a distinct file ID and

a certain number of copies. Then, the user chooses a version, downloading pieces of different copies. If a downloaded piece is a corrupted part from the changed file, the entire download file will be corrupted. The hash corruption is the name we give to this technique to pollute files.

Note that hash corruption mechanism does not necessarily break hash functions such as MD4, MD5 and SHA-1. It takes advantage of the way that the P2P systems use these functions to create file IDs. As an example, the FastTrack [5] network uses an algorithm called uuhash [17], which uses only some pieces of the content file to generate the ID. Consequently, a different file with the same ID can be created by changing the parts of the file which were not used as input of the hash function. Another way to corrupt the ID would be to change a P2P client so that instead of correctly computing the ID for a file, it associates a corrupted file with the ID of the target file.

4. EVALUATION METHODOLOGY

This Section describes the methodology used in our study. We developed an event-driven simulator able to reproduce the common aspects of pollution dissemination in current P2P architectures. The main aspects and assumptions adopted in this simulator are described in Section 4.1. In Section 4.2 we present our metrics and the parameters we vary in our simulations.

4.1 Simulator Description

Since our study aims at understanding the dissemination of polluted content, we concentrate on evaluating only the peers which have the file under consideration. The simulation begins with a constant number of peers, 10% of the maximum number of peers; that is, it begins with 5,000 peers, since the maximum population is set to 50,000. The file has 200 versions, and each version has a number of copies. Initially the number of copies per version follows a Zipf distribution with α coefficient 1. Each of the initial peers has one copy of one version of the file, and at any time a peer can have at most one copy of the file. We consider the system to consist of the peers that currently have a copy of the file. The peers request the file, and thus enter the system, following a Poisson process with a rate of 0.1 peers per step. When a peer request a file, it receives the entire file on the next simulation step. For simplicity, we do not model transfer time. When a peer downloads a file, we assume that if the file is not immediately deleted the user share the polluted content. If the file is deleted the user leaves the system and do not return.

In P2P systems, after a peer requests a file it usually receives a list of versions of the file, and a number of sources for each version. The requesting peer chooses one of the versions to download according to the popularity of the version. For instance, if 50% of the available copies are of a single version, the probability that the peer chooses this version is 0.5. Since we are not evaluating the search mechanism, we do not describe the P2P overlay topology for the peers, and we assume that the peer which requests a file always finds all versions and copies of the file. Our system model captures, to a first order, the most relevant tradeoffs to evaluate the impact of pollution on P2P systems. We assume that all peers in the system share their content and we do not model peer availability.

4.2 Metrics and Parameters

In order to evaluate pollution dissemination in P2P systems we define **polluted content dissemination** at a given time as the percentage of active peers in the system whose copy of the file is polluted at that time.

The parameters that we vary in order to evaluate polluted content dissemination are summarized next:

- **Incentive for cleaning (IC)**: we call the incentive for cleaning, IC, all the mechanisms deployed in P2P systems which aim at avoiding pollution. We do not create or evaluate any specific incentive mechanism for cleaning polluted content. So, IC corresponds to the probability that a user immediately deletes a downloaded polluted file.

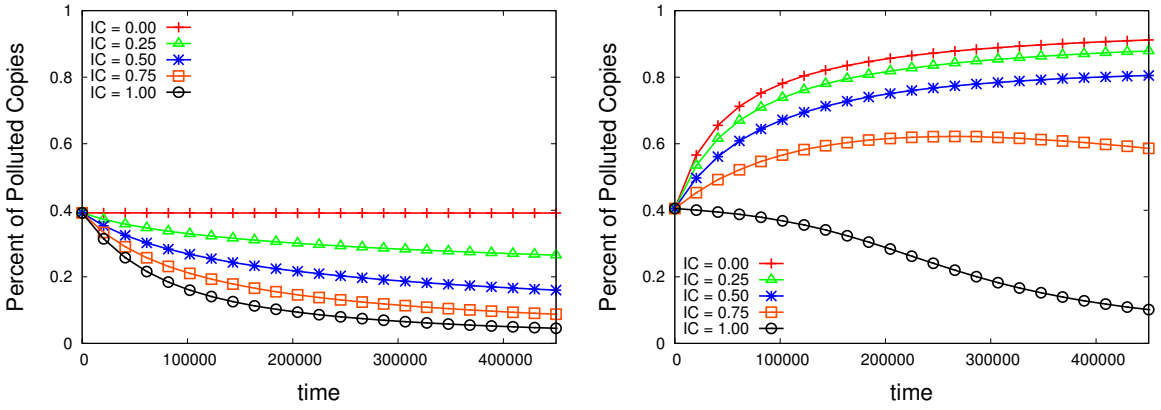
- **Hash weakness (HW)**: in order to evaluate the entire class of algorithms which generate file IDs, we define the hash weakness, HW, as the percentage of a file content that can be corrupted without changing its file ID. For instance, using uuhash[17] to generate the hash code of files of 5 MB and 600 MB, the hash weaknesses of these files would be 88% and 99.5% respectively.

- **Number of download sources (NS)**: when a peer searches for a file it receives a list of file versions. Each version has a number of sources from which it can be downloaded. We call NS, the maximum number of simultaneous sources a file can be downloaded. We evaluate dissemination of polluted content for different values of NS for the hash corruption mechanisms. Note that, NS has no impact for decoy insertion since all sources of a given version are either polluted or non-polluted. The NS affects the pollution by hash corruption, where we can have both polluted and non-polluted content in a same version.

5. RESULTS

This Section presents the results most relevant to the dissemination of polluted content in P2P networks. All the graphs show the polluted content dissemination: the Y-axis shows the percentage of copies in the system that are polluted, and the X-axis shows the elapsed time. We present results for a system which begins with 5,000 peers and 40% of the copies polluted. We did experiments with different values for the numbers of initial peers and percentage of polluted copies and we found qualitatively similar results. The percentage of polluted copies used on the simulations is typical of real systems [9]. The simulation ends when the system population achieves 50,000 peers (45,000 downloads). Each simulation result is an average of 10 runs. With a confidence level of 95%, the results differ from the mean by 10% at maximum.

The dissemination of polluted copies for the decoy insertion mechanism is shown in Figure 1-a. In these experiments, some versions are randomly chosen to be polluted, and the distribution of the initial number of copies of these polluted versions is uniform over the versions. Each line shows the number of polluted copies for different values of incentive for cleaning (IC). As expected, even with a small incentive for cleaning, the percentage of polluted copies on the network decreases along the time. Note that when users



(a) decoy insertion mechanism of pollution

(b) hash corruption mechanism of pollution. $HW = 90\%$

Figure 1: Dissemination of polluted copies varying the incentive for cleaning (IC). $NS = 10$

do not delete their polluted content, the percentage of corrupted copies does not change. This is because the probability that a user entering the system requests a polluted version of the file is equal to the percentage of copies that are corrupted, and this maintains the ratio between polluted and non-polluted copies of the file. When the incentive for cleaning increases, the percentage of content that is polluted significantly decreases. If the users have the probability of 0.25 for cleaning their corrupted content ($IC = 0.25$), the percentage of polluted copies in the network decrease 35% when simulation ends.

Figure 1-b shows the corresponding graph for the hash corruption mechanism. In this scenario, all versions initially have the same percentage of polluted copies, so that popular versions have more polluted copies. We assume that a user downloads content from 10 sources simultaneously ($NS = 10$) and that initially a polluted file has 90% of its data corrupted by hash corruption mechanism ($HW = 90\%$). We can observe that, even if users have an incentive for cleaning ($IC > 0$), the number of polluted files increases over time for the hash corruption technique. This effect occurs because the download is made from 10 different sources, and all versions have polluted copies.

Comparing the mechanisms of decoy insertion and hash corruption, we note the pervasiveness of the hash corruption mechanism. Such effectiveness occurs because the download is made from various different sources (10 in our experiments) and all versions have polluted copies. For instance, if just one of the 10 download sources provides a corrupted copy, the user may download a corrupted piece from that source, polluting the downloaded file. Pollution caused by decoy insertion can be sizeably reduced by increasing the incentive for cleaning, whereas with the hash corruption mechanism the effect of increasing the incentive for cleaning is smaller. We can see in Figures 1-a and 1-b that while in decoy insertion mechanism the percentage of pollution decreases in 65% by the end of simulation for $IC = 0.5$, this percentage increases 101% using the hash corruption technique.

Figure 2 shows results varying the hash weakness (HW). In order to evaluate just the effect of hash weakness we assume that users do not delete the polluted content ($IC = 0$) and the number of simultaneous download sources is 10 ($NS = 10$). The curves show that even a small hash weakness has a strong impact on the system. For instance, with 30% ($HW = 30\%$) of the file polluted, we see the percentage of polluted copies rise above 65% in the end of simulation. This happens because when the hash weakness increases, the probability of a polluted source uploads corrupted data also increases.

Finally, Figure 3 shows how pollution dissemination is affected by increasing the number of sources (NS) from which the file is downloaded, under the hash corruption mechanism for pollution. As expected, the larger the number sources from which a file is downloaded, the higher the probability of part of the file be polluted. Note that even when the download is made from a single source, the hash corruption mechanism maintains the proportion of polluted copies constant, just as the decoy insertion mechanism does.

6. CONCLUSION AND FUTURE WORK

In this paper we investigated the effectiveness of peer incentives for cleaning corrupted files in reducing the dissemination of polluted content, considering two potential strategies for polluting P2P systems: decoy insertion and hash corruption. We show that the hash corruption spreads pollution faster than decoy insertion, and even if the users have a high incentive to clean polluted content¸ the hash corruption mechanism is an efficient way of disseminating pollution. Moreover, we show that hash corruption can be more pervasive when the download is done from a high number of different sources.

Directions for future work include study and analysis of incentives mechanisms, and the implementation of new features in the simulator: (1) peer availability, where peers join and leave the system; (2) popularity of sources, which consists of applying different probabilities for the choice of download sources. We will investigate how pollution dis-

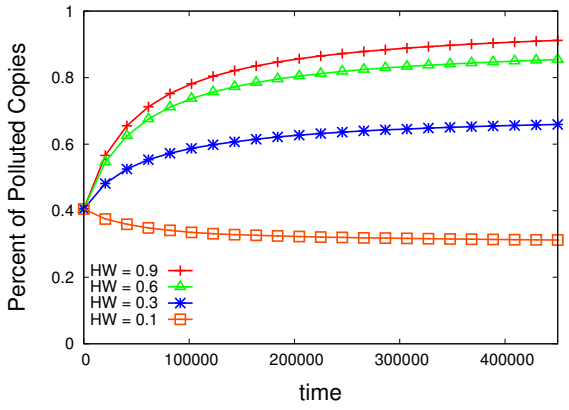


Figure 2: Dissemination of polluted copies by hash corruption mechanism varying hash weakness (HW). $IC = 0.0$ and $NS = 10$

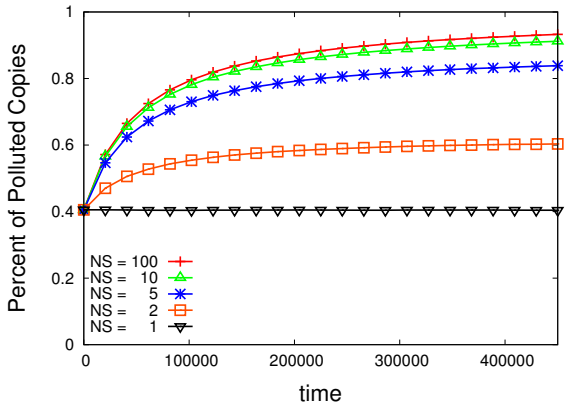


Figure 3: Dissemination of polluted copies by hash corruption mechanism varying the number of simultaneous download sources (NS). $IC = 0.0$ and $HW = 90\%$

semination is affected by reputation mechanisms like Credence [19] and how a network with a high percentage of polluted copies can be healed. Furthermore, we will propose an analytical model to extend our work.

7. ACKNOWLEDGMENT

This work was developed in collaboration with HP Brazil R&D.

8. REFERENCES

[1] F. Benevenuto, J. I. Junior, and J. Almeida. Quantitative evaluation of unstructured peer-to-peer architectures. In *Proc. of IEEE First International WorkShop on Hot Topics in Peer-to-Peer Systems (Hot-P2P'04)*, Volendam, The Netherlands, 2004.

[2] BitTorrent. http://bitconjurer.org/bittorrent/.

[3] N. Christin, A. S. Weigend, and J. Chuang. Content availability, pollution and poisoning in file sharing peer-to-peer networks. In *Proc. of ACM E-Commerce Conference*, Vancouver, Canada, June 2005.

[4] eDonkey. http://www.edonkey2000.com/.

[5] Fasttrack. http://www.fasttrack.com.

[6] Gnutella. http://www.gnutella.com.

[7] KaZaA. http://www.kazaa.com.

[8] J. Kurose and K. Ross. *Computer Networking: A Top-Down Approach Featuring the Internet.* Addison-Wesley, 2005.

[9] J. Liang, R. Kumar, Y. Xi, and K. W. Ross. Pollution in p2p file sharing systems. In *Proc. of IEEE Infocom*, Miami, FL, USA, March 2005.

[10] Napster. http://www.napster.com.

[11] Overpeer. http://www.overpeer.com.

[12] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips. The bittorrent p2p file-sharing system: Measurements and analysisng. In *Proc. of IPTPS*, Ithaca, NY, USA, February 2005.

[13] A. R. *The Evolution of Cooperation.* Basic Books, 1984.

[14] Retspan. http://www.retspan.info.

[15] S. Saroiu, S. D. Gribble, and H. M. Levy. Measurement and analysis of spyware in a university environment. In *Proc. of the 1st Symposium on Networked Systems Design and Implementation (NSDI)*, San Francisco, CA, March 2004.

[16] S. Saroiu, P. Gummadi, and S. Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems. In *Proc. Multimedia Computing and Networking 2002 (MMCN '02)*, San Jose, CA, USA, January 2002.

[17] UUHash. http://en.wikipedia.org/wiki/UUHash.

[18] Viralg. http://www.viralg.com.

[19] K. Walsh and E. G. Sirer. Fighting peer-to-peer spam and decoys with object reputation. In *Proc. of the Third Workshop on the Economics of Peer-to-Peer Systems (p2pecon)*, Philadelphia, PA, USA, August 2005.

[20] W. Yu, C. Boyer, S. Chellappan, and D. Xuan. Peer-to-peer System-based Active Worm Attacks: Modeling and Analysis. In *Proc. of IEEE International Conference on Communications (ICC 2005)*, Seoul, Korea, May 2005.