DETECTING MALICIOUS HOSTS IN THE PRESENCE OF LYING HOSTS IN PEER-TO-PEER STREAMING

Xing Jin[*] *S.-H. Gary Chan*[*] *W.-P. Ken Yiu*[*] *Yongqiang Xiong*[†] *Qian Zhang*[*]

[*] Department of Computer Science
The Hong Kong Univ. of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
{csvenus, gchan, kenyiu, qianzh}@cs.ust.hk

[†] Microsoft Research Asia
Beijing Sigma Center, No. 49, ZhiChun Road
Haidian District, Beijing 100080, P. R. China
yqx@microsoft.com

ABSTRACT

Current peer-to-peer (P2P) streaming systems often assume that hosts are cooperative. However, this may not be true in the open environment of the Internet. In this paper, we discuss how to detect malicious hosts (e.g., with attacking actions and abnormal behavior) based on their history performance. In our system, each host monitors the performance of its neighbor(s) and reports this to a server. Based on the reports, the server computes host reputation with hosts of low reputation being malicious. A problem is that hosts may lie by submitting forged reports to the server. We hence formulate the reputation computing problem in the presence of lying hosts as a minimization problem and solve it by the traditional Levenberg-Marquardt algorithm. Simulation results show that our scheme can efficiently detect malicious hosts with high accuracy.

1. INTRODUCTION

A peer-to-peer streaming system, characterized by a single source and multiple recipients, overcomes limitations in traditional server-based systems in terms of bandwidth and user scalability. It eliminates the need for powerful severs by distributing the load among peers. Some current P2P streaming systems have been shown to be able to serve up to thousands of peers with acceptable quality of service (QoS) [1].

Most of the P2P systems work on the assumption of truthful cooperation among peers. However, in the open environment of the Internet, some participating hosts may not cooperate as desired. They may be selfish and unwilling to upload data to others, or they may have abnormal actions such as frequently rebooting which adversely affect their neighbors. More seriously, some hosts may launch attacks to disrupt the service or distribute viruses in the overlay network. We call all these uncooperative, abnormal or attacking behavior *malicious actions* and the associated hosts *malicious hosts*.

In this paper, we study how to detect malicious hosts in a P2P streaming system. We are interested in identifying the malicious hosts that tamper the streaming quality of some of their neighbors. Such malicious actions can be detected based on host history. If each host is periodically monitored by its neighbors and the monitoring reports are analyzed by a server, hosts with malicious behavior can be easily detected. However, a problem is that a host may lie about the performance of its neighbors in the monitoring reports. Such a host can affect the evaluation as it desires.

To address this problem, we define two numerical metrics between 0 and 1 for each host. One is host *reputation* that indicates to what extend the host is malicious; a reputation value of 0 means that the host is fully malicious while a reputation of 1 suggests that the host is not malicious at all. Another metric is host *credibility*, which indicates to what extend we should believe the report sent by the host. We assume that hosts taking malicious actions and lying action are independent. We consider that a small fraction of hosts in the streaming system are malicious or lying, and host behavior is consistent over time.

There is a detector server to collect monitoring reports from hosts. The problem of computing host reputation based on the reports in the presence of lying is formulated as a minimization problem and solved by the traditional Levenberg-Marquardt algorithm [2]. Hosts with low reputation are then evaluated as malicious. We have done simulations to evaluate our scheme. The results show that it can detect malicious hosts with low false positive and false negative rates.

We briefly review previous work on P2P reputation as follows. Trust-Aware Multicast (TAM) computes a level of trust for each host according to their behavior and builds a multicast tree according to the trustworthiness of hosts [3]. P-Grid trust model uses a DHT-like (Distributed Hash Table) distributed information access system to manage host reputation [4]. However, they have not considered host lying in the monitoring reports. EigenTrust also uses a DHT network to

This work is supported, in part, by the Area of Excellence in Information Technology of the University Grant Council (AoE/E-01/99), Competitive Earmarked Research Grant of the Research Grant Council (HKUST6156/03E) in Hong Kong and the Innovation and Technology Commission of the Hong Kong Special Administrative Region, China (GHP/045/05).

store and access host reputation [5]. It requires each host to iteratively evaluate others' reputation according to reports from third-party hosts. A problem is that the final reputation values may take a long time to converge. We formulate the reputation computing problem as a minimization problem, which can be efficiently solved by traditional optimization methods. Malicious Detector Algorithm (MDA) considers lying problem in reputation computing [6]. For each transaction between two hosts, both hosts need to submit a report to evaluate the transaction. If the two reports are different, the transaction is evaluated as suspicious. MDA computes host credibility according to the suspicious transactions and further computes reputation based on the reports and host credibility. Our scheme takes a different approach to compute host credibility, by formulating it as a minimization problem. It only requires one participant of a transaction to submit a report and hence can efficiently reduce the monitoring cost.

The rest of the paper is organized as follows. In Section 2 we discuss how to detect malicious hosts in the presence of lying hosts. We present in Section 3 some simulation results, and conclude in Section 4.

2. DETECTING MALICIOUS HOSTS

In this section, we first discuss some malicious actions that can be detected by peer-based monitoring, and then study how to compute host reputation in the presence of lying hosts.

2.1. Malicious Actions in P2P Streaming

We are interested in the malicious actions that can be detected through the monitoring of the past performance of hosts. Such actions include

- Attacking actions: Some hosts intend to degrade system performance and disrupt the service. Their attacking actions include

  (1) *Eclipse attack*: In an overlay network, if an attacker controls a fraction of the neighbors of a legitimate host, it can "eclipse" the legitimate host by dropping or rerouting messages and adversely affect proper overlay operation [7]. To detect an Eclipse attack, a host may report the number of connections its parents are setting up, since an attacker usually has significantly larger in-degree and out-degree as compared to the average values in the system [7].

  (2) *Resource-consuming attack*: A host may request as much data as possible from as many peers as possible. To detect this, a host may report the amount of resources its neighbor is consuming.

  (3) *Distributing corrupt data*: A host may send out corrupt data that do not conform to the stream format. To detect it, a host may report the amount of corrupt data its parent sends out with respect to the total amount of data the parent sends out.

  (4) *Attracting peers without serving them*: A host may advertise a large amount of network resources (such as bandwidth) to attract many peers but does not serve them. This can be detected by host reporting the amount of data a host sends to each of its children.

- Abnormal behavior: Some hosts do not intend to attack the system, but their actions adversely affect other peers and degrade the service. These abnormal actions include

  (1) *Frequent joining/leaving*: The host joins and leaves the system frequently. This can be detected by its neighbor reporting the joining and leaving moments of the host. The average system time of the host can then be computed.

  (2) *Free riding*: A host may only download data but not share them with others [8]. This can be detected by host reporting the amount of download and upload data of a neighbor.

Though the above malicious actions may be detected by peer monitoring, it becomes an issue if some hosts lie in their reports. How to detect malicious hosts in this setting is not trivial.

2.2. Design Overview

We use a server to collect monitoring reports generated by hosts. Define a *submission period* as the time interval between two consecutive reports of the same host. The performance of a host is recorded by its streaming neighbors, and the reports are sent to the server at the end of the period. The server then summarizes all the reports to compute host reputation to identify malicious behavior.

There are many ways to use the evaluation results. In one case, if host A wants to set up a connection with another host B, A first queries B's reputation. If B has low reputation (i.e., a potential malicious host), A blacklists B and does not connect to it for a while. In another case, if some host is considered as malicious, the server broadcasts this information to other peers so that they may block it.

2.3. Reputation Computing

For ease of illustration and brevity, we consider detecting one of the above malicious actions, say, distributing corrupt data. It will be clear that our approach is general enough to extend to other malicious actions. Let G be the set of hosts in the streaming system, and $|G|$ be the number of hosts in G. The children of a host i record the amount of corrupt data and total data received from i in each period. In the tth period, define $Corrupt(i, j, t)$ and $Total(i, j, t)$ as the amount of corrupt

and total data received from i to j, respectively. Let R_i be the reputation of i and C_{ij} be i's view on j's credibility. Furthermore, let C_j be the consensus credibility of j, which is the final credibility of j based on the reports from all the hosts.

The reputation of host i is given by the average over all the reported reputation values on i, according to

$$R_i = Average\Bigg\{C_{ix} \times \left(1 - \frac{Corrupt(i,x,y)}{Total(i,x,y)}\right), \quad \forall x \in G \; and \; y \in valid \; periods\Bigg\}, \tag{1}$$

where valid periods restrict the average over the latest reports and exclude outdated reports.

When a host x have submitted a report on i, the credibility of x with respect to i, C_{ix}, is updated as

$$C_{ix} = Average\left\{1 - \left|R_i - \left(1 - \frac{Corrupt(i,x,y)}{Total(i,x,y)}\right)\right|\right\}.$$

Therefore, the closer to i's reputation the report from x is, the higher the credibility of x is.

In the ideal case, the credibility of a host x from any host's viewpoint should be the same, which is x's consensus credibility. We hence seek the set of $\{C_x, \forall\ x \in G\}$ to minimize the following objective function:

$$f = \sum_{\forall i \in G} (R_i - R_i^*)^2,$$

where R_i^* is computed as Equation (1) by replacing C_{ix} by C_x.

This problem can be solved by the Levenberg-Marquardt algorithm, which provides a numerical solution to the mathematical problem of minimizing a sum of squares of several functions [2]. The algorithm requires an initial guess for the unknown variables (i.e., C_x, $\forall\ x \in G$ in our problem). In many cases, the algorithm can quickly find a solution even if the initial guess is very far from the optimum. We use $C_x = \frac{1}{|G|} \sum_{\forall i \in G} C_{ix}$ as the initial guess.

As we consider most hosts in the system are non-malicious, a host is evaluated as malicious if its reputation value is smaller than the average reputation by a certain threshold and an absolute threshold value.

3. ILLUSTRATIVE NUMERICAL RESULTS

In this section we present simulation results on our scheme. We randomly put 5000 hosts into the network and form a streaming overlay as follows. Each host randomly selects multiple hosts as its parents, and a host can have at most 10 children. A host may also dynamically change some of its parents. We summarize the main simulation parameters in Table 1.

Table 1. Simulation Parameters.

Name	*Description*	*Default*
Group size ($\|G\|$)	the number of hosts in the system	5000
Maximum/Average number of children	the maximum/average number of the children of a host on the streaming overlay	$10/5$
Average child life	the average duration of a streaming connection between two hosts (in the unit of submission period)	25
Monitoring time (t)	in the unit of submission period	100
Malice ratio (α)	the number of malicious hosts divided by the total number of hosts	25%
Lie ratio (β)	the number of lying hosts divided by the total number of hosts	25%

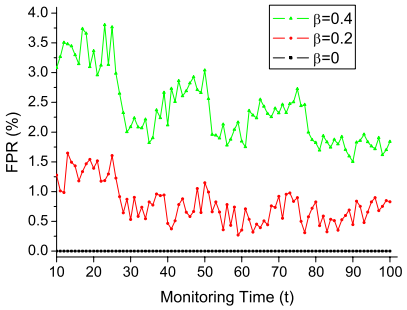
We simulate the example of distributing corrupt data. A non-malicious host distributes a negligible amount of corrupt data, while a malicious host distributes corrupt data with a probability uniformly distributed between $[0.4, 1]$. Furthermore, a host may lie in its reports to the detector server. A lying host lies with a a probability uniformly distributed between $[0.4, 1]$, while an honest one never lies. Note that the behavior of malice and lying are independent. A host is evaluated as malicious if its reputation is smaller than the average reputation value of all the non-leaf hosts and a threshold 0.7.

We define two metrics for evaluation:

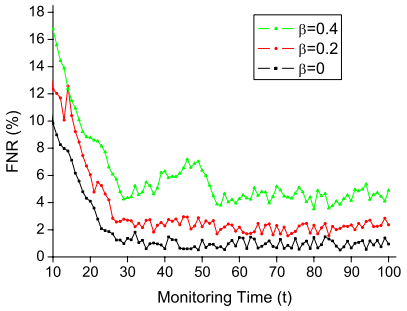
- *False positive rate (FPR)*: defined as the number of non-malicious hosts evaluated as malicious divided by the total number of non-malicious hosts.
- *False negative rate (FNR)*: defined as the number of malicious hosts evaluated as non-malicious divided by the total number of malicious hosts.

Figure 1 shows FPR and FNR values versus monitoring time given different lie ratios. In the first several dozen periods, both FPR and FNR are large. They quickly decrease with the monitoring time. It shows that in the starting stage the credibility and reputation values are skewed. The monitoring duration does not need to be long (about 30 periods) to achieve satisfactory accuracy. The larger the lie ratio, the larger FPR and FNR. In Fig. 1(a) FPR is kept below 3.5% in the steady stage, even with 40% lying hosts. In the best case of $\beta = 0$, FPR is always 0. On the other hand, we achieve much larger FNR values as shown in Fig. 1(b). This is partially due to the small evaluation threshold we set, because it is more important to protect non-malicious hosts than to detect malicious ones.

Figure 2 shows FPR and FNR values with different malice ratios. Similarly, they are large in the starting stage and become much smaller in the steady stage. We note in Fig. 2(a) that the scheme achieves higher FPR when the malice ratio is low. When $\alpha = 0$, the FPR value is almost the largest. This is because our judgment is based on the average reputation value. With small malice ratios, a large portion of hosts are non-malicious and have similar reputation values, which

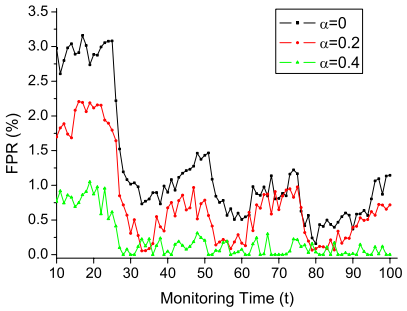


(a) FPR;

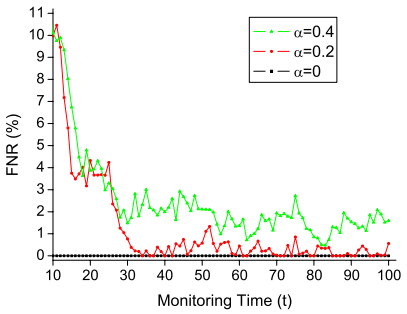


(b) FNR.

Fig. 1. Performance with different lie ratios.



(a) FPR;



(b) FNR.

Fig. 2. Performance with different malice ratios.

are also close to the average. In this case, a small perturbation to the evaluation may lead to incorrect judgment. While in a system with large malice ratios, the gap between the average reputation and the reputation of a non-malicious host is large, therefore non-malicious hosts are unlikely to be evaluated as malicious.

4. CONCLUSION

Most proposed P2P streaming systems assume that hosts are cooperative. However, this may not be true in the Internet. In this paper, we study how to detect malicious hosts based on their history. Each host monitors its neighbors and periodically reports their performance to a server. Considering that some hosts may lie by submitting forged reports, we study how to compute host reputation based on the reports and host credibility. We formulate the problem as a minimization problem and solve it by the Levenberg-Marquardt algorithm. Simulation results show that our scheme can detect malicious hosts with low error.

5. REFERENCES

[1] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "CoolStreaming/DONet: A data-driven overlay network for efficient live media streaming," in *Proc. IEEE INFOCOM'05*, March 2005.

[2] "Levenberg-marquardt Method." http://mathworld.wolfram.com/Levenberg-MarquardtMethod.html.

[3] S. Jun, M. Ahamad, and J. Xu, "Robust information dissemination in uncooperative environments," in *Proc. IEEE ICDCS'05*, 2005.

[4] K. Aberer and Z. Despotovic, "Managing trust in a peer-2-peer information system," in *Proc. ACM CIKM'01*, 2001.

[5] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The EigenTrust algorithm for reputation management in P2P networks," in *Proc. WWW'03*, 2003.

[6] L. Mekouar, Y. Iraqi, and R. Boutaba, "Detecting malicious peers in a reputation-based peer-to-peer system," in *Proc. IEEE CCNC'05*, 2005.

[7] A. Singh, M. Castro, P. Druschel, and A. Rowstron, "Defending against Eclipse attacks on overlay networks," in *Proc. SIGOPS EW'04*, 2004.

[8] E. Adar and B. A. Huberman, "Free riding on Gnutella." Tech. Rep., HP, http://www.hpl.hp.com/research/idl/papers/gnutella/gnutella.pdf, 2000.