5-2008

A Dynamic Trust Management Scheme to Mitigate Malware Proliferation in P2P Networks

Xuhua DING
Singapore Management University, xhding@smu.edu.sg

Wei YU
Texas A & M University

Ying PAN
Singapore Management University

DOI: https://doi.org/10.1109/ICC.2008.310

A Dynamic Trust Management Scheme to Mitigate Malware Proliferation in P2P Networks

Xuhua Ding
Singapore Management University
xhding@smu.edu.sg

Wei Yu
Texas A&M University
weiyu@cs.tamu.edu

Ying Pan
Singapore Management University
panying@gmail.com

***Abstract*—The recent surge of peer-to-peer (P2P) networks consisting of thousands of of hosts makes them a breeding ground for malware proliferation. Although some existing studies have shown that malware proliferation can pose significant threats to P2P networks, defending against such an attack is largely an open problem. This paper aims to develop the countermeasure that can effectively mitigate the malware proliferation while preserving P2P networks' performance. To this end, we propose a dynamic trust management scheme based upon localized trust evaluation and alert propagation which prevents innocent peers from downloading files from infected peers. Our analysis and experimental results show that our approach can effectively reduce the malware proliferation rate.**

***Index Terms*—P2P systems, Malware proliferation, Dynamic trust.**

I. Introduction

In recent years, the explosive surge of Peer-to-Peer (P2P) networks has been astounding. While P2P networks facilitate the content distribution, they offer a breeding ground for malware proliferation due to their daily thousands of file downloads. After being initially inserted by malicious peers, malware can propagate along with the file downloading path. Consequently, innocent peers incidentally downloading those infected files will become new sources of infection. In this manner, malware propagates itself to a large number of peers within a short period, posing a significant threat to P2P networks.

Since content distribution such as file sharing is the main venue of the malware proliferation in P2P networks, those network-based anti-malware systems, such as packet filter firewalls or traffic anomaly monitors, are ineffective. Although deploying application based firewall in the P2P network could mitigate the attack, this approach can drastically downgrade the system performance due to the huge volume of traffic in the P2P network. Therefore, it is desirable to develop the malware defense based on the P2P network itself.

We highlight the subtle yet important differences among our study, P2P worms [1], [2] and P2P pollution [3], [4]. The threat that we deal with is generalized malware, including viruses, spyware or even worms, which propagates itself by taking advantage of the content distribution of P2P networks. P2P-based worms studied in [1], [2] is a specific type of worm that exploits the common vulnerabilities among the P2P client software and hunts its preys by scanning other P2P peers via various means, such as P2P overlay network topology. Pollution attacks intend to disrupt the P2P file sharing services by alluring peers to download corrupted files, e.g., mp3 files which cannot be played [5]. Obviously, the pollution tactics can be used for malware propagation. However, corrupted files are always detected by *every* user, whereas malware is unfortunately not, since detecting and removing malware depend on many factors, including the capability of peers' anti-malware software and users' security awareness etc.

To effectively mitigate the malware proliferation in the P2P network, it is critical to design schemes that prevent innocent peers downloading files from "bad" peers harboring contagious files. In this paper we propose a new countermeasure called *dynamic trust management* scheme based upon localized trust evaluation and propagation. In particular, we adopt a trust mechanism among all peers. Every peer locally manages her trusts to neighboring peers based on the quality of previously downloaded files. A peer always downloads files from those with the highest trust. If malware is detected, a peer reduces her trust on the involved file provider. A trust degradation greater than a prescribed threshold will trigger a trust alert propagation to related peers, which then make appropriate adjustments as well. Our analysis and simulation results demonstrate that this approach can significantly reduce the malware proliferation rate.

II. A Dynamic Trust Management Scheme

A. Design Rationale

We consider an adversary who attempts to propagate malware to peers in a P2P network for malicious purposes, e.g., stealing user identities, using the infected hosts to form botnets [6], [7]. The adversary may join the P2P network using several legitimate identities and prepares a large number of malware-carrying files to share with others. An innocent peer becomes infected if she downloads and opens an infected file. Once the malware is activated in one peer, it may infect other files in the victim's file directory. To detect the malware, a user may conduct anti-malware scanning on her file downloaded directory. Malware could be detected by an anti-malware software with negligible false positives and false negatives. Once being detected, the malware is removed locally. We assume the existence of a PGP (pretty-good-privacy)-like public key infrastructure such that the integrity of peer communications are ensured.

The basic idea of our scheme is to incorporate a dynamic trust management within the P2P network. The fact that peer A has a trust on peer B reflects the degree of A's expectation

on the probability of a correct file downloading from B. The more trust A has on B, the higher the expectation. Each peer in our system trusts a set of peers called *trustees* and is trusted by another set of peers called *trusters*. A peer's trustee set and truster set may have an overlap. A peer selects the source of her file download based on her trusts on the file providers. Without loss of generality, we now use peer A as an example to provide the overview of our scheme.

During the registration or boot-up process, A obtains a list of peers, from which she is able to select a group of peers as her trustees. Other peers who are aware of A's arrival may add A to their trustees. When A needs to download a file, she broadcasts her query to all her trustees. From those who reply positively, A selects the one with the highest trust value, say B, as the download source. Once the download interaction completes, A adjusts her trust value on B. For example, a failed download or an infected file may cause A to degrade her trust on B. If the trust degradation is greater than a prescribed threshold, an alert of trust adjustment is triggered and broadcasted to A's trusters. Depending on their trusts on A and trusts on B, A's trusters may respond to the alert in different ways. A's trusters make necessary downgrade of their trusts on B, which may likely trigger another alert. In this manner, the alert of the degradation of B's credibility is proactively propagated to a subset of B's trusters.

Despite the apparently straightforward mechanism of our proposed scheme, several key issues need to be considered.

- The trigger of a trust propagation depends on the threshold of trust degradation. A too low threshold would cause unnecessary network communication overhead while a too high threshold would possibly result in a delayed propagation and consequently more infected peers.
- A peer's response to an alert is crucial to the effectiveness of the defense system as well. An overly sensitive peer may unnecessarily expand the range of a propagation while a reckless peer may cause negligence of the malware alert.
- Like a route update in routing protocols, a trust propagation algorithm must converge quickly. In other words, an alert should not cause peers to repetitively send and receive the same event.

B. Detail Design

Consider a P2P network of n peers denoted by $V = \{v_1, v_2, \cdots, v_n\}$. We define the trust value to be a quotient number in $[-1, -1] \cup [0, 1]$. If peer v_i has -1 trust value on peer v_j, v_i distrusts v_j. Trust value 0 implies a neutral trust and trust value 1 represents the highest trust. Peer v_i's trustees are organized in a *trustee table*, denoted as $T_i = \{\langle v_j, t_j \rangle | v_j \in V, t_{ij} \in [0, 1]\}$, where t_{ij} is the value of v_i's trust on v_j; and v_i trusters are organized in a *truster set*, denoted as $S_i \subset V$. Our design complies with the well-known *trust decay rule* [8]: a person tends to have much more trust on those referred by his/her immediate friends than on those referred by friends of friends. More generally, the trust value decreases proportionally to the "social" distance between two peers.

In our system, there are a system-wide parameter ϕ and a function $F(x)$ used by all peers.

- ϕ: $0 < \phi < 1$, the trust variation threshold for triggering a trust alert.
- $F(x) : [0, 1] \rightarrow [0, 1]$, the trust value computation function, where $0 \leq x \leq 1$ is the download success rate with respect to a peer.

In the following, we introduce four core procedures of our proposed scheme.

1) Initialization: When a new peer (say v_i) wants to join the P2P network, the initialization task consists of two parts. One is to initialize v_i's trust values on a set of peers and the other is for other peers to assign their trust values on v_i.

The new peer v_i is given a list of peers as its trustee candidates by a bootstrapping server. Along with the addresses of those candidates, the system provides other individual information, such as their online histories, nationalities, file depository size, interests. Based on the information and her own security profile, v_i initializes her T_i by selecting those peers from the candidate list and assigning them positive trust values. If the size of T_i is not large enough, she may invoke a *referring* process. A referring process allows v_i's trustees to introduce more trustee candidates to v_i. To start a referring process, v_i selects the trustees, say v_j, with the highest trust value as her reference peer. v_j then sends her trustee table T_j to v_i. For an entry $\langle v_k, t_{jk} \rangle$ in T_j, v_i computes $t_{ik} = t_{ij} \times t_{jk}$, where t_{kj} is v_i's trust on v_j. If t_{ik} is large enough to satisfy v_i's security profile, she inserts $\langle v_k, t_{ik} \rangle$ into T_i.

Once T_i is finalized, v_i sends all peers in T_i a message which encloses v_i's individual information and notifies them that v_i has trust on them. Accordingly, peers in T_i insert v_i to their truster sets and may also assign their trust on v_i according to v_i's profile.

2) File Search: To download a file f, peer v_i sends a *file query* to all her trustees in T_i. If positive responses are received, v_i selects the trustee with the highest trust as the download source. If no positive responses are received, v_i sends a *help query* to all her trustees. Each v_i's trustee recursively searches for f with the help of their own trustees respectively. Without loss of generality, suppose that $v_j \in T_i$ and $v_k \in T_j$, v_k replies to v_j positively. v_j then sends a positive reply to v_i which contains the identity of v_k and the trust value t_{jk} on v_k. We denote v_k as a *referred trustee* recommended by v_j. v_i's potential trust value on v_k is computed as $t_{ik} = t_{ij} \times t_{jk}$. Among all referred trustees, v_i downloads the file directly from the one with the highest potential trust value.

3) Local Trust Update: For each trustee in T_i, v_i maintains a download transaction record. After downloading files from a peer v_j, v_i reevaluates her trust on v_j according to the *success rate* (denoted by ρ_{ij}) of interactions with v_j, which is calculated as the fraction of success downloads among all downloads from v_j. A failure is defined as the occurrence of a malware infected file downloaded from v_j. v_i then computes

$t'_{ij} = F(\rho_{ij})$. If $t_{ij} - t'_{ij} > \phi$, v_j replaces t_{ij} with t'_{ij} as the new trust value, and executes the propagation procedure to alert other peers as shown in the next subsection.

4) Propagation of Trust Alert: An alert is flagged once a peer detects a significant drop of trust on another peer. A propagation of alert warns other peers who have not noticed the trustworthiness downgrade. In this proactive way, there will be a smaller possibility for innocent peers being exposed to malware. Consequently, the proliferation of malware is suppressed. Peer v_i's alert message is of the following format: (ID, P, Δ, d), where ID uniquely identifies the alert, defined as the binary concatenation of *time* and v_i's id; P is the id of the subject peer of this alert, e.g. v_j's id; Δ is the magnitude of the trust downgrade, e.g $t_{ij} - t'_{ij}$; d is the maximum number of hops this alert is allowed to travel.

Peer v_i broadcasts the message to all peers in her *truster set*, (instead of her trustee set). A peer receiving an alert from v_i adjusts the relevant trust value accordingly and may forward this alert to her trusters. In general, when a peer v_j receives an alert (ID, v_m, Δ, d) from v_k, she responds in the following manner: if (ID, v_m) is duplicated with a previously received alert or $v_m \notin T_j$, this alert is dropped immediately. Otherwise, v_j sets $t_{jm} = t_{jm} - t_{jk}\Delta$. If $t_{jk}\Delta < \phi$ or $d = 1$, v_j terminates her response procedure. Otherwise, she prepares for forwarding the alert by setting $d = d - 1$ and $\Delta = t_{jk}\Delta$. The other two fields are not changed. The new alert is sent to all v_j's trusters. Note that the propagation converges quickly since both Δ and d decrease after one hop. Due to the length limit, we skip the discussion on the timing of arrivals of the same alerts sent from different sources. Our detailed algorithms include the setting up of a timeout window and the management of a message buffer to accommodate the variance of alert propagation delays due to different routes. More details are presented in a full version of the paper.

C. Discussions

Our scheme has several salient features summarized as follows.

1) Push-based Trust Adjustment: In existing trust or reputation schemes [9], [10], a peer needs to *pull* relevant data from either a group of peers or a centralized server in the network. By contrast, a peer in our scheme updates her trust on other peers in a more timely fashion. An alert of a service degradation from a peer is *pushed* to related peers who promptly make their own appropriated responses. For defending against poisonous file spreading [11], both *pull* and *push* approaches have relatively equivalent effects. However, our push-based approach outperforms its counterpart when dealing with malware propagation. This is because malware detection suffers from a false positive rate (a.k.a miss rate). *Push*-based approach enables one successful detection to be propagated to multiple peers who might fail the detection. Compared to pull-based schemes, our scheme is more proactive and prevent more peers from contacting infected files.

2) Negative Alert: In our scheme, only a negative trust variation triggers an alert propagation. One might argue that a positive trust adjustment should be propagated as well. Nonetheless, propagating positive alerts may becomes a vulnerability exploited by an adversary. For instance, a collusion of adversaries are able to take advantage of it and convince other honest peers to trust a malicious peer. With negative trust propagation, faked alerts only lead to a DoS attack on a well-behaved peer. Considering the nature of file sharing and the fully distributed P2P networks, such DoS attacks in a individual user lack strong motivations.

3) Localized Trust Management: All trust value computations in our scheme are executed locally and do not require peer interactions during the computation. Therefore, our scheme has a better computational efficiency than those global or local reputation schemes which require interactions with other peers to compute a reputation. Our scheme does not involve any global algorithms, nor requires a system-wide consistency among peers' trusts. Those global algorithms such as voting schemes, reputation schemes, usually become the target of attacks and may consequently be misused. Once such a scheme is manipulated by an attacker, its adversarial effect is even worse than not applying it. In our scheme, each peer independently manages her own trusts to others. By not relying on any centralized service or system-wide mechanisms, our scheme minimizes both the incentive of attacks and the vulnerabilities exploited by an adversary.

4) Socialized Trust: Our trust management conforms with the recent trend of incorporating the social network concept into P2P networks [12]. It is more efficient and natural for users with common interests to share files with each other. We envision that relatively stable socialized connections are less risky and more effective for user to search and download files than volatile random connections. Moreover, the socialization provides the agreeable and robust infrastructure to the trust management, since it is not effective to employ trust for volatile random associations.

5) Content Availability: Content availability is a critical performance metric to evaluate a P2P network's service. Although our trust-based approach deters the proliferation of malware, it does not hinder the dissemination of benign contents. Note that most anti-malware software have negligible false negatives. An honest user's file service will not be affected. Moreover, even if a malicious user launches a DoS attack on a user, the impact of her faulty alarms depends on both the number of her trusters and her own credibility. Therefore, the malicious user has to provide better services to maximize the impact, which to some extent offsets the adversarial effect of her attacks.

III. Analysis

To characterize malware proliferation over the P2P network, we adopt the epidemic dynamic model for the theoretical analysis [13] and carry out an analysis using discrete time to derive recursive formulae for the malware proliferation.

For $i \geq 0$, let $I(i)$ be the number of infected peers in the P2P network at the beginning of the i^{th} period. Suppose that each peer launches λ downloads during the i^{th} period. Let α_i be the

probability of contacting an infectious peer for each file query during the i^{th} period and β be the probability of every peer successfully identifying a downloaded malware. Therefore, for each query in the i^{th} period, a peer has $\tau_i = \alpha_i(1-\beta)$ probability of being infected. Let $\bar{\tau}_i = 1-\tau_i$. We assume that there are initially I_0 malicious peers uniformly distributed in the systems. Thus, $I(0) = I_0$ and $\alpha_0 = I_0/n$.

We define a random variable $X_{i,j} \in \{0,1\}$ which represents the event that the j^{th} uninfected peer, $j \in [1, n-I(i)]$, becomes infected by the end of the i^{th} period. In particular, if the j^{th} uninfected peer is finally infected, $X_{i,j} = 1$; otherwise, $X_{i,j} = 0$.

For a native P2P network, we have $\mathbb{P}(X_{i,j} = 0) = (1-\tau_i)^{\lambda}$ and $\mathbb{P}(X_{i,j} = 1) = 1-(1-\tau_i)^{\lambda}$. Thus, we have the mean of $X_{i,j}$: $\mathbb{E}(X_{i,j}) = 1-\bar{\tau_i}^{\lambda}$. The average number of newly infected peers during the i^{th} duration is $\mathbb{E}(\sum_{j=1}^{n-I(i)} X_{i,j}) = (n-I(i))(1-\hat{\tau_i}^{\lambda})$. According to our assumption of the uniform distribution of infected peers, we have $\alpha_i = \frac{I(i)}{n}$. Thus, the number of infected peers in the $(i+1)^{th}$ duration is

$$I(i+1) = I(i) + (n-I(i))(1-\bar{\tau_i}^{\lambda}), \tag{1}$$

where $\bar{\tau_i} = 1-(1-\beta)\alpha_i$. Approximately, we have

$$\begin{cases} \alpha_i = \frac{I(i)}{n}, \\ I(i+1) = I(i) + \lambda(n-I(i))\alpha_i(1-\beta). \end{cases} \tag{2}$$

Since λ mainly depends on the user behaviors, we treat it as a constant. Therefore, the effectiveness of a security mechanism dealing with malware proliferation relies on the magnitude of increasing $\bar{\alpha_i}$, or equivalently reducing α_i.

In essence, our dynamic trust management scheme is to reduce α_i by preventing peers from contacting known malicious ones. To simplify the modeling, we suppose that a trust alert is propagated to m peers in average. In addition, we assume that every user will postpone all removals of malicious peers detected during the i^{th} period to the end of the period. We define a random variable $Z_{i,j}$ to represent the event that the j^{th} infected peer, $j \in [1, I(i)]$, is detected by a peer during one download. In particular, if the j^{th} infected peer is detected during the i^{th} period, $Z_{i,j} = 1$; otherwise, $Z_{i,j} = 0$. Therefore, we have $\mathbb{P}(Z_{i,j} = 1) = \beta\alpha_i$ and $\mathbb{P}(Z_{i,j} = 0) = 1-\beta\alpha_i$.

Let Y_i be the average number of infected peers identified in the i^{th} period. According to our model simplified as $m\lambda$ Bernoulli trials, we have $Y_i = m\lambda\beta\alpha_i$. Therefore,

$$\alpha_{i+1} = \frac{I(i) - m\lambda\beta\alpha_i}{n - m\lambda\beta\alpha_i}. \tag{3}$$

Moreover, since peers share their alerts, we have approximately $\mathbb{P}(X_{i,j} = 1) = \lambda\tau_i(1-\beta/n)^{(m-1)\lambda}$. Thus, $\mathbb{E}(\sum_{j=1}^{n-I(i)} X_{i,j}) = \lambda\tau_i(n-I(i))(1-\beta/n)^{(m-1)\lambda}$. Then, we have

$$I(i+1) = I(i) + \lambda(n-I(i))\alpha_i(1-\beta)(1-\beta/n)^{(m-1)\lambda}. \tag{4}$$

Approximately, we have

$$\begin{cases} \alpha_{i+1} = \frac{I(i)-m\lambda\beta\alpha_i}{n-m\lambda\beta\alpha_i}, \\ I(i+1) = I(i) + \lambda(n-I(i))\alpha_i(1-\beta)(1-\frac{m\beta\lambda}{n}). \end{cases} \tag{5}$$

Compare Equation 5 against Equation 2, we conclude that our proposed trust propagation scheme reduces the growth rate of $I(i)$ by both decreasing the contacting probability α_i from $\frac{I(i)}{n}$ to $\frac{I(i)-m\lambda\beta\alpha_i}{n-m\lambda\beta\alpha_i}$ and decreasing the missing rate of malware detection from $(1-\beta)$ to $(1-\beta)(1-\frac{m\beta\lambda}{n})$). Also, we conclude that a larger m and a larger β are always favorable to mitigate the malware proliferation whereas increasing λ may not. Note that our focus in this analysis is to model the proliferation of malware by counting the increment of infected peers. It is easy to revise the above model to consider the case where a peer recovers from an infected state to a clean state.

IV. Experiments

We simulate a small scale P2P network with an emphasis on the effectiveness of our scheme, without tuning various parameters mentioned in Section II and Section III. Our simulated network consists of 100 randomly connected peers. Each node has no more than 10 neighbors. Based on the previous studies on content distribution [14], [15], we implement a Zipf content distribution model. Since the number of distinct files usually is around 10 times of the number of peers [16], we initialize 1000 distinct files with various copies in the system. Only about 10% peers have more than 100 files while most peers have much less files to share.

Among all peers, we randomly select 10 of them as malicious peers whose files are marked as malware. We assign every honest peer v_i a random number $\beta_i \in [0.6, 1]$. If v_i downloads an infected file, she will be marked as an *infected peer* with a probability $1-\beta_i$. This probability is to simulate her local capability in malware detection. We do *not* simulate the local file infection. In other words, when an innocent peer downloads a file containing malware, other existing files in her storage directory are not contaminated.

In our experiments, we compare the malware proliferation rate in the same system with different countermeasure schemes: (1) *Baseline scheme.* In this scheme, the P2P network offers no security countermeasures to resist malware proliferation. Each peer randomly selects the download source and relies on her own capability to detect malware. (2) *Scheme with trust only.* In this scheme, the network has a basic trust scheme. Every peer downloads files according to the trust relations and locally manages her trusts on neighbors based on the quality of the historical interactions. However, *no* trust alert is propagated. (3) *Scheme with trust and propagation.* In this scheme, our proposed dynamic trust management scheme is adopted.

We measure the total number of infected peers, including the initial 10 malicious peers, to evaluate the malware proliferation rate. After every 50 downloads, we take a snapshot of the system to collect the number of infected peers. The results for three schemes are shown in Figure 1.

Due to the experimental randomness, the gradient of the lines may vary from experiment to experiment. Nonetheless, the curve of the baseline scheme always stays on the top whereas the curve of trust propagation always stays at the bottom and remains horizontal almost for every experiment.

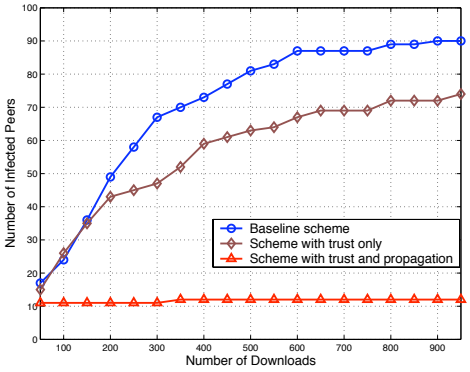

Fig. 1. A comparison of malware proliferation rates in three schemes

We make two important observations from our experimental results. First, using anti-malware software alone cannot successfully prevent malwares from spreading in P2P networks. Those users with weaker malware detection capability will become the victims and the accomplices of malware spreading. Second, our proposed scheme is able to effectively suppress the malware spreading, since the peers virtually share their anti-malware detections. Although our experiment set is relatively small compared to those P2P networks in real-world, we argue that the scale of the network does not overturn our conclusion. Specifically, few large networks in practice are randomly connected, since the network interactions always exhibit locality patterns. Therefore, a large network can be treated as a congregation of a number of small scale networks. The capability of the proposed scheme actually depends on the locality pattern, rather than the overall system size.

V. Related Work

The modeling of worm propagation and pollution attacks in P2P networks is studied in [2], [3], [4], [11] etc. For instance, Yu *et al.* in [2] modeled the P2P network based worm attacks and revealed that a worm can propagate more rapidly over the Internet by using the P2P network. There are a few research efforts for developing countermeasures against worm propagation and pollution attacks. For example, Zou *et al.* in [1] investigated the worms that taking advantage of P2P weakness and proposed several countermeasures based on individual P2P peers to carry out detection and post-detection mitigation.

Our work is also related to the reputation and trust system research [9], [10]. These approaches are either based on centralized management or on a tacit assumption that trust or reputation of a node reflects the majority nodes' trust in the whole network. However, for a large scale P2P network, it is infeasible to obtain votes from a majority of the whole system due to the performance constraints. On the contrary, a peer only contacts a small fraction of the whole population. An adversary may take advantage of this limitation to abuse these reputation and trust systems. Our work is also related to a recent research [12] that incorporates the social-network concept into P2P networks to facilitate content discovery and downloading.

VI. Conclusion

In this paper, we address the issue of defeating the malware proliferation over the P2P network. We propose a dynamic trust management scheme based upon local trust evaluation and propagation. Our proposed scheme has several salient features such as push-based trust adjustment, localized trust management, etc. Both the analysis and the experimental evaluation results demonstrate the effectiveness of our scheme in terms of low malware proliferation rate. Our study helps to safeguard P2P users from malware proliferation when distributing files.

References

[1] L. D. Zhou, L. T. Zhang, F. Mcsherry, N. Immorlica, M. Costa, and S. Chien, "A first look at peer-to-peer worms: threats and defenses," in *In Proceedings of the 4-th International Workshop on Peer-To-Peer Systems (IPTPS)*, Ithaca, NY, February 2005.

[2] W. Yu, C. Boyer, S. Chellappan, and D. Xuan, "Peer-to-peer system-based active wor attacks: modeling and analysis," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Seol, Korea, May 2005.

[3] P. Dhungel, X. Hei, K. W. Ross, and N. Saxena, "The pollution attack in p2p live video streaming: measurement results and defenses," in *Proceedings of Sigcomm P2P-TV Workshop*, Kyoto, Japan, August 2007.

[4] R. Thommes and M. Coates, "Epidemiological modeling of peer-to-peer viruses and pollution," in *Proceedings of IEEE INFOCOM*, Phoenix, AZ, April 2006.

[5] J. Liang, R. Kumar, Y. Xi, and K. W. Ross, "Pollution in p2p file sharing systems," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2005, pp. 1174–1185.

[6] P. R. Roberts, *Zotob Arrest Breaks Credit Card Fraud Ring*. http://www.eweek.com/article2/0,1895,1854162,00.asp.

[7] R. Naraine, *Botnet Hunters Search for Command and Control Servers*. http://www.eweek.com/article2/0,1759,1829347,00.asp.

[8] A. Jøsang, E. Gray, and M. Kinateder, "Analysing topologies of transitive trust," in *Proceedings of the 1th International Workshop on Formal Aspects in Security and Trust (FAST)*, Pisa, Italy, August 2003.

[9] E. Damiani, D. C. Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A reputation-based approach for choosing reliable resources in peer-to-peer networks," in *Proceedings of the 9th ACM conference on Computer and communications security (CCS)*, Washington, DC, November 2002.

[10] S. Marti and H. Garcia-Molina, "Limited reputation sharing in p2p systems," in *Proceedings of the 5th ACM Conference on Electronic commerce (EC)*, New York, NY, May 2004.

[11] R. Kumar, D. D. Yao, A. Bagchiz, K. W. Ross, and D. Rubenstein, "Fluid modeling of pollution proliferation in p2p networks," in *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, St. Malo, France, June 2006.

[12] J. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Losup, D. Epema, M. Reinders, M. Steen, and H. Sips, "Tribler: a social-based peer-to-peer system," in *Proceedings of the 5-th International Workshop on Peer-to-peer Systems (IPTPS)*, Santa Barbara, CA, February 2006.

[13] D. J. Daley and J. Gani, *Epidemic Modeling: an Introduction*. Cambridge University Press, 1999.

[14] R. R. Korfhage, *Information Storage and Retrieval*. Wiley, 1997.

[15] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Proceedings of IEEE Internation Confernece on Multimedia Computing and Networking*, 2002.

[16] M. Karakaya, I. Korpeoglu, and O. Ulusoy, "Gnusim: A general purpose simulator for gnutella and unstructured p2p networks," Department of Computer Engineering, Bilkent University, Tech. Rep., 2005.