CPSC557
Sensitive Information in a Wired World

Reputation Management in P2P Systems

Pradipta Prometheus Mitra
Department of Computer Science, Yale University

December 12, 2003

Contents

Chapter 1

Introduction

Peer-to-peer (P2P) systems have gained enormous popularity in a very short timespan. Numerous systems have been deployed and are being used right now. Napster was the first P2P system to be immensely successful (though it had a centralized server for searching). More decentralized P2P systems such as Gnutella, Kazaa have been developed subsequently. Advanced data structures such as distributed hash tables have later been used to build P2P systems. Such efforts have resulted in systems like CAN, Chord and Tapestry. Recently, BitTorrent has been the first P2P system to implement a scheme that encourages good behavior through incentive.

The widespread use of P2P has created the need for trust management for these systems. The open nature of these systems makes it easy for entities with malicious intent to join the system and try to undermine it. For P2P to be successful in a world where trust and careful handling of sensitive information is growing to be a top priority, it is important that these systems develop a stong scheme for trust management while retaining their open nature.

This report will contain 6 chapters, including this one. In chapter 2, we look at how P2P has been used in the past and present, and what the future prospects of these systems are. We specially look at the prospect of P2P arising as viable business model in the digital world. We try to establish the need for a trust management system for P2P. We would also see why trust management in P2P will be different from that in centralized system architectures and how the uniqueness of P2P calls for a different type of trust management, namely distributed reputation management.

In chapter 3, we establish the system model which will be the basis for the subsequent chapters. We spell out assumptions about the network structure. The issues of document authenticity is dealt with. The possible threat models in P2P systems are discussed and the parameters and metrics used to measure the efficiency and effectiveness of reputation management systems are established.

Chapter 4 studies the first model for reputation management, which is Limited Reputation Management. We discuss local and voting schemes. Simulation results are presented to show the expected performance of these schemes. The next chapter introduces EigenTrust, a global reputation management scheme. The algorithm is outlined, and we compare it to limited reputation management. The algorithms presented in these chapters are from works done by the Stanford Database Group (chpaters will contain exact references).

Identity in Peer-to-peer systems is the topic of chater 6. We show how anonymity, the much prized chracteristic of most P2P systems, cause an almost insurmountable problem for reputation management. Possible ways to tackle this problem would be discussed.

The last section concludes the paper by summarizing the results and posing some research problems.

At the end of most chapters, we pose some questions and problems related to the material presented in the chapter.

Chapter 2

P2P as a Business Model and the Need for Managing Trust

Inspite of its immense popularity and the widespread interest surronding it, P2P has the reputations of being no more than a file sharing system. This reputation of P2P started with Napster, and it must be said that to date, most P2P systems are being used for disseminating files between the users. Most of the files shared are multimedia files and a huge number of them pirated. In October 2002, The Recording Industry Association of America and the Motion Picture Association of America sent a joint letter to the universities in the United States urging them to monitor and control use of P2P software on university campuses[10]. It is not clear how such a system would evolve into a feasible business model.

The argument against peer-to-peer systems is mainly that any content that would have widespread demand can be supplied by conventional business models and doesn't need P2P systems for it. And the most important use of P2P, that of file sharing has hardly any revenue model and therefore no business model. It is true that so far there has been no substantial business models based on the P2P architectures. However, some recent works show that there might be some possibilities in these directions and it might not be impossible for P2P to emerge as a strong business model in the digital world.

We discuss these briefly here:

- Currently myCIO.com is providing a service to combat viruses using P2P distribution. Using P2P, virus update information can be distributed much more efficiently that was previously possible. As users share virus alerts, the alerts get through the system in a matter of minutes rather than hours or days if it were to be done by a centralized entity (and possibly manually). It can also be used for distributing updates and patches for antiviruses effectively. This is a scenario where trust management might be very useful. It is important that a virus disseminator cannot undermine the system by sending false alerts and thereby destroying the effectiveness of true alerts.

- P2P systems can be used for reliable backups. These systems, however might be much more closed in nature and might not have the same dynamics that more open P2P systems have.

- Penn State is developing a P2P system named LionShare for distributing works of academia through a global and secure network. Though this system itself might not have a direct revenue model, similar systems might be used for accessing other sorts of data that might create a business proposition.

- In [11], the authors propose that a business model for P2P can be built based on the "Virtual Community" concept proposed by Rheingold in [12][1]. Though they admit that it's difficult to remove the central server completely from such "communities" if they are to function effectively, such a system might still possess a lot of P2P like characteristics and hence share many of the problems regarding trust management.

We may conclude that though a complete business model for P2P systems is yet to evolve, labeling P2P as a mere "piracy machine" might be quite wrong as well. Moreover, a robust trust management scheme might be important for P2P to take the next step a sucessful business model. The trust management schemes used for more centralized systems are usually based on cryptography, and are more about certificate generation and policy. However, in a P2P system, where there might not be any centralized server for generating certificates and no clear notion of policy, these schemes are not very relevant. "Reputation management" has evolved as the major trust management scheme for P2P systems and it can be briefly described as trusting (or distrusting) a peer based on it's past behavior. In the next chapter, we develop a systems model for P2P as the basis for discussing reputation management.

[1] Amazon and EBay are examples of this

Chapter 3

Reputation Management: System Model

The anonymity and open nature of P2P systems is something that must be taken into account while designing reputation management schemes for them. It is important that these features are retained as much as possible while still building a reliable system for reputation management. In [2], five design considerations are presented:

1. The system should be self-policed. That is, no central authority should manage or enforce the system.
2. Anonymity should be maintained.
3. New-comers should not get any profit. This shall take away the incentive for malicious peers to obtain new indentifiers.[1]
4. The system should have minimal overhead.
5. The system should be robust to collective of mailicious peers.

3.1 System Configuration

3.1.1 Assumptions

These are the main assumptions regarding P2P systems that we consider:

[1]This is a complicated issue. Treating new-comers just as malicious peers might make it difficult for new-comers to build a reputation. Actually, it might be useful to sacrifice some of the previous considerations [atleast partially] to make joining the system easier. This issue will be dealt with in chapter 6.

1. The system is composed of n nodes arranged in an overlay network.

2. Nothing is assumed about the graph structure. It can be unstructured, as in Gnutella, or highly structured, as in CAN or Chord. However, most simulations done so far have been on Gnutella like random structures.

3. When a node needs a resource, it sends a *query* to some or all of its peers in the system. It collects responses in a *Response Set*, and then selects a provider from which to retrieve the resource. In all current systems the provider is selected from the response set in a random fashion.

4. After downloading the file(or using the resource), each peer verifies its validity.

3.1.2 Document Authenticity

In the presence of malicious nodes, it is important to be able to detect the authenticity of a resource. Unfortunately, checking the validity of a document is highly domain specific. The problem of determining whether a document is authentic has been referred to in the literature as the *document authenticity* problem [5]. This clearly is not the problem of file corruption while transmission, a problem well known and adequately dealt with by redundancy techniques. The problem with document autheticity is that it cannot always be calculated programmatically. It often requires user interaction, such as reading a document or listening to a music file. Moreover, it may not be possible for a certain user to verify the authenticity of a resource correctly. A alternative approach might be to have a central authenticating entity verifying documents (either by online verification or some form of cryptographic digital signature). However, that clearly isn't possible always and also isn't a good fit with the open, decentralized nature of P2P systems.

In Reputation Management literature, the problem of document authenticity is not dealt with at all. Studies usually assume that there is a global authenticity function. That is, each user can verify a authenticity of a file (by whatever means) and assign a 0/1 value to it.

3.2 The Threat Model

There are two types of threats in a P2P system: selfishness and maliciousness.

3.2.1 Selfishness

The first kind of misbehavior is selfishness. Selfishness can manifest itself in different forms:

Free Riding Accessing resource in the network but sharing none. A study on Gnutella show that almost 70% of Gnutella users share no files, and nearly 50% of all responses are returned by the top 1% of sharing hosts [6]. However, P2P systems tend to perform well in these situations and it isn't a very big problem.

Uploading at Slow Rates Selfish peers share files, but don't put in enough effort to serve peers interested in those files. They save their bandwidth for their own downloading purposes. Though theoretical schemes for combating this sort of behavior have been around for some time, BitTorrent [7] is the first widely deployed P2P system that incorporates such a framework.

The way BitTorrent achieves Parreto Efiiciency is described in [8]. BitTorrent uses a combination of Choking and Unchoking algorithms to decide which peers to upload to and which peers not to. It uses a rolling 20-second average, so the algorithm is quite responsive to changes in a peer's behavior [2]. It also uses "Optimistic Unchoking" to allow uploads to peers that it didn't have previous communication with [or to connections not used for a long time]. This correlates closely with the first step in the iterative version of Prisoner's Dilemma.

3.2.2 Malicious Behavior in P2P

Following are the main types of maliciousness discussed in the literature [1, 2, 5]:

Individual Malicious peers They always provide an inauthentic file when selected as a download source. They don't collude with others, simply lie in the network for their own gain. They give a bad opinion of everyone else[3].

Malicious Collectives Malicious peers collude in providing high reputation values for each other and low for all other nodes.

[2] This makes sense in that problem domain, but unfortunately not in the domain of malicious peers spreading inauthentic files

[3] In [2], the model in slightly different. Here the malicious peers will give high opinion of peers they download from and find the files to be inauthentic.

Malicious Collectives with Camouflage Provides inauthentic files in $f\%$ of cases that it is selected.

Malicious Spies Themselves provide good files but try to increase the reputation of other malicious peers. Also known as "front nodes".

A set of files are targeted by the malicious nodes for subversion. This set in known as the *subversion set*. The percentage of nodes in the network is given by π_B. Each file has an equal probability of being in the subversion set, specified by the parameter p_B and it is assumed that there is no correlation between a files popularity and p_B. Studies also usually assume that there is no other types of attack in the system, such as DOS attacks.

3.3 Goals and Metrics

The main objective of a reputation system is to reduce the number of documents the user must look at before finding a valid one. This is the same as minimizing the number of times the authenticity function has to be invoked. This and other metrics are defined formally below:

Verification Ratio Verification ratio is the most important metric in evaluating a reputation management scheme and is defined as

$$r_V = \frac{\sum_{i=1}^{n} V_i}{q_{succ}} \tag{3.1}$$

Where, V_i is number of file verifications done for files supplied by node i. Hence the summation is simply the total number of verifications done in the system[4].
q_{succ} is the number of successful queries.

r_V is also referred to as the *efficiency* of the system.

Load Load on a node is defined as

$$l_i = \frac{V_i}{q_{tot}} \tag{3.2}$$

V_i is same as before and q_{tot} is the total number of queries in the system. The average load l_G is an average taken across the set of all well-behaved nodes, G.

$$l_G = \frac{\sum_{i \in G} l_i}{n_G} \tag{3.3}$$

Here, $n_G = |G|$.

[4] This is clearly same as the number of downloads

Message Traffic How a certain reputation scheme affects the message traffic in the system. A specific metric is relative message traffic, described more clearly in 4.4.

3.4 Questions

1. Even if we assume that resources can be verfied correctly, is a binary value of authenticity accurate? Specially in the case of multimedia files, this scheme doesn't capture the intermediate cases. For example, low resolution pictures, music files with last 5 seconds absent, Black-and-white print of a color film can't be called "inauthentic". Is there any way to differentiate between "low quality" and "inautheticity"?

2. How justified is the assumption of no correlation between a files popularity and the probablility of its being targeted for subversion? How would results change if the probability is assumed to be proportional to popularity?

Chapter 4

Limited Reputation Management in P2P

4.1 Why do we need "Limited" Reputation Management?

The most widely studied method of trust management for P2P systems, as we have mentioned in the previous chapters, is reputation management. There are two important and unfortunately conflicting design goals that we strive to achieve in these systems. The first in robustness of the system, we want it to be relaible and stable in presence of malicious peers. On the other hand, we want these systems to be scalable. If P2P systems are to be used for more seious applications than file sharing, both robustness and scalability are factors of great importance.

The conflicting nature of these two goals comes from the fact that while robustness can only be ensured if reputation is calculated from a global perspective, that does not give us a scalable solution. If global information need to be collected and precessed, the amount of information grows atleast quadratically with number of nodes. Limited reputation management seeks to find the best of both worlds. They achieve scalability by allowing single nodes to communicate with only a small number of other nodes, thereby ensuring that overhead increases only linealy with increase in network size. The schemes discussed here are from [1]. The paper provides simulation results to prove the claim that these limited reputation management schemes do well in the presence of mailicious adversaries. However, no substantial theoretical underpinning has been provided so far, so its difficult to tell if these algorithms will display robust behavior in all possible settings.

We shall consider two schemes as presented in [1], namely

1. Local Reputation Management
2. Voting Reputation Systems

4.2 Local Reputation Management

Simply stated, in this scheme, no information is shared between peers about reputation. Nodes only depend on their own past experience with other nodes. The scheme has no scalability problems at all, but suffers from inefficiency. It is more useful as a basis for developing Voting Reputation Systems.

Main features of this system:

1. In this system, each node maintains statistics on how many files it has verified from each peer and how many of those were authentic. Each peer's reputation rating is calculated as the fraction of verified files which were authentic. This results in a rating ranging from 0 to 1. Naturally, 0 means none of the files verfied were authentic and 1 would mean all of the verified files were authentic.
2. When a node gets back the response set for a query, it uses these ratings for selecting the peers from which to download the file from.

4.2.1 Peer Selection in Local Reputation Systems

The ranking can be used for peer selection in 2 different ways:

Select Best Simply selects the the response from the node with highest rating. If the file turns out to be invalid, it choses the next one and so on.

Weighted Selection Select Best has the adverse effect of overloading the best nodes in the system. Weighted selection procedure, probabilistically selects the file to fetch. The probablity of a node being chosen is proportional to its rating. For example, if nodes i and j provide replies to q and $R(q,i) = 0.1$ and $R(q,j) = 0.7$ then j is 7 times more likely to be picked.

Simulations show that the weighted method distributes load better without degrading efficiency much.

4.2.2 ***Friend-Cache*** in Local Reputation Systems

In Select-best method a node maintains an ordered list of most reputable nodes it knows. This list is known as the friend-cache and has a maximum size FC. The friend-cache can be used for optimizing the network performance considerably. By sending queries directly to nodes in the cache before querying the network in the usual fashion message traffic can be reduced considerably.

4.3 Voting Reputation Systems

This scheme extends the local reputation system by considering opinions of other peers in the selection stage. When a node q receives a set of responses to a query, it contacts a set of nodes Q, for their opinions of the responders. The final rating for each responder p_r is calculated by:

$$p_r = (1 - w_Q)R(q,r) + w_Q \frac{\sum_{v \in Q} R(q,v)R(v,r)}{\sum_{v \in Q} R(q,v)} \tag{4.1}$$

Important aspects of this scheme are:

1. For each "voter", the opninion of the voter is weighted by the reputation of that voter.
2. The final rating is a combination of local experience and peer responses. These two values are combined using quorumweight w_Q.
3. When $w_Q = 0$, the voting reputation system becomes the local reputation system.
4. Select-best and weighted procedures apply to voting reputation systems as well.

4.3.1 Voter Selection

For a scalable solution a node should only query a limited number of other nodes, but select those nodes judiciously to get useful information. In [1], two ways of selecting "voters" are proposed:

Neighbor-voting Asking opinions from ones neighbors in the overlay network. Number and Identity of voters remain relatively constant over time.

Friend-voting Asking peers from whom one has fetched files and who have proven to be reliable. The friend-cache can be used. Number of friends depend on FC, the maximum size of the cache.

Friend voting is clearly better, as a node having the bad luck of being surrounded by malicious neighbors may get a really bad response if it uses neighbor voting. Simulation results support this intuition as we shall see later.

4.3.2 Initial Reputation Rating and Selection Threshold

An initial reputation rating ρ_0 is assigned to new nodes, or nodes that a peer never had interaction with. Peers also use a selection threshold ρ_T. All responses from a node whose reputation ranking is below this threshold is automatically discarded and never considered for downloading[1] (thus decreasing the probabililty of downloading files that almost sure to be fake) .

4.4 Simulation Results for Limited Reputation Sharing

Simulation results presented here are from [1]. We shall only succintly present here the main findings, not actual quantitative results (which can be found in the paper referred).

Along with verification ratio and average load, Relative Message Traffic has been studied as an important metric in [1]. It is defined as

$$MT_{rel} = \frac{N_{friend}}{N_{flood}} \tag{4.2}$$

N_{friend}: No of messages when the friend-first method is used
N_{flood}: No of messages when normal flooding is used for queries.

It is also assumed that connectivity follows Power-law distribution and so does number of copies of each file. These are both reasonable assumptions [9].

For whitewashing(malicious peers can change their identities freely) $p_0 = 0$, otherwise $p_0 = 0.3$.

[1] This threshold does not apply for new nodes, files will be downloaded from them even if $\rho_0 < \rho_T$

(Default) fraction of malicious nodes $p_B = 0.3$.
Main findings from simulation results:

- Friend voting does better than neighbor voting. Neighbor voting suffers quite a bit in whitewashing conditions, but friend voting manages to do well.

- If well-behaved nodes offer bad-copies unknowingly, matters complicate. However, simulations show that as long as the probability of that happenning is small, the results don't change markedly.

- The size of the friend-cache doesn't affect the results much after the size exceeds certain values (around 15 for a 1000 node netowrk).

- Front nodes affect the efficiency heavily, specially if quorumweight w_Q is high (more importance is given to voters than the node's own experience).

- Values in the range of 0.1 to 0.2 for w_Q give best performances.

- Friend voting continues to perform reasonably for values of π_B upto 0.4.

- Experiments show non-intuitive results for load. Select-best yields a lower load for the most highly loaded node than weighted selection. However, the balancing is better for the weighted case, as expected.

- The Friend-first query scheme produced very good results. It resulted in upto 85% less message traffic.

4.5 Questions

1. What happens if some nodes are "good" in all ways, except that they use select best procedure for peer selection? What is the effect on load distribution? Can such behavior be detected?

2. In voting reputation system, the same reputation matrix is being used for judging files sent by peers and opinions sebt by peers. This does not seem to be an adequate model in the presence of "front nodes". Does it help to have seperate file reputation ranks and opinion reputation ranks? What about incorporation of opinion information into the reputation rank (as it stands now, only file authenticity contributes to the rank)?

3. Often in a network like these, there are some peers who are mostly "providers", while others are mostly "consumers". In the friend-voting system, a peer would ask information to nodes that are providers. In a situation where most of these peers don't access resources very often(are mainly servers), this might not work well. Is it possible to ask to other "consumers"? Should the information provided by a peer also be weighted by the number of files(or resources) that peer has accessed? How can that information be verified, if at all?

4. Values of ρ_0 and ρ_T (initial reputation rating and selection threshold, respectively) are fixed constants. What about allowing dynamic values for these parameters, so that nodes can update them according to their experience over time?

5. Should not time required for a new good node to build a reputation be considered as an important metric?

Chapter 5

Global Reputation Management in P2P: EigenTrust

In limited reputation management, only local knowledge is used for reputation management. This can lead to reduced performances, specially for high values of π_B (fraction of malicious nodes). A trivial way to extend the voting reputation system to a global scheme would be to have nodes ask all other nodes for their reputation information. However, this clearly has various difficulties. First, a node may not know all other nodes in the system. Secondly, this would raise the overhead in the system to impractical levels. Moreover, the reputation rankings for each node does not necessarily converge because of quorumweight associated to the vote.

In [2], a different approach is proposed for global reputation calculation. The algorithm is known as the EigenTrust algorithm. We shall present the basic features of EigenTrust and an outline of its mechanisms.

5.1 Local and Normalized Trust

The local trust value a node i assigns to a peer j is defined as

$$s_{ij} = sat(i,j) - unsat(i,j) \tag{5.1}$$

$sat(i,j)$ and $unsat(i,j)$ are, respectively, the number of satisfactory and unsatisfactory downloads by i from j. To ensure that all values are between 0 and 1, this local trust value is normalized as follows:

$$c_{ij} = \frac{max(s_{ij},0)}{\sum_j max(s_{ij},0)} \tag{5.2}$$

This method of normalization gives a good probabilistic interpretation of trust, but has some difficulties. It is undefined if $\sum_j max(s_{ij}) = 0$. Also trust becomes relative, so we can only compare between to nodes, but not have an absolute interpretation.

5.2 Outline of EigenTrust

While calculating the reputation of peek k, peer i might choose to ask its acquaintances about k. The reputation found by this method woud be

$$t_{ik} = \sum_j c_{ij} c_{jk} \tag{5.3}$$

In matrix notation this might be expressed as $C^T \vec{c_i}$. To incorporate second level friends (acquaintances of acquaintances), we would calculate $(C^T)^2 \vec{c_i}$. If this continues, for some sufficiently large value of n, $(C^T)^n \vec{c_i}$ would converge for all i, and thus we would have a global trust assigned to each peer.

Had there been a centralized server, it could collect normalized trust values from the peers, calculate the global reputation matrix and disseminate them. Unfortunatley, we don't have a centralized server to do the job. In EigenTrust, this problem is solved by using a DHT (distributed hash table). The DHT is used to assign *score managers* to calculate the reputation for a peer. Multiple score managers are assigned to calculate the reputation of the same peer to make the system secure.

5.3 Performances and Problems

- Simulation results show that EigenTrust often converges within resonable number of cycles. For a 1000 node network, the algorithm converges after 100 query cycles.

- As a global scheme is used, the load distribution in calculating reputation is better than in limited schemes (where nodes tend to query better nodes for reputation ranking of another peer).

- EigenTrust does extremely well even in the presence of high number of malicious nodes. It exhibits low verification ratio even when $\pi_B = 0.7$.

- It does reasonably well in the presence of malicious collectives.

- Requires much more communication overhead than limited reputation systems.

- Uses DHT as its main data structure for secure score management. This makes EigenTrust usable only with complex P2P systems such as CAN and Chord.

- The case where well-behave nodes can share bad files unknowingly hasn't been handled in EigenTrust adequately either.

5.4 Questions

1. How does the setting change if we allow peers to declare their confidence in the authenticity of the file they are sharing? In this case, if a peer is not sure that the file it is sharing is authentic, it may declare a 0.60 confidence in the authenticity of that file. Might this form of "honesty" help distinguish between malicious nodes and nodes inadvertently sharing invalid files?

2. Can EigenTrust be extended to simpler P2P systems such as Gnutella?

Chapter 6

The Identity Dilemma

Anonymity is one of the most important distinguishing features of P2P systems. However, in the presence of malicious nodes, it creates a possiblity of various attacks on such systems. Anonymity allows a backdoor for malicious nodes to enter and exit the system at their will.

Problems arising from the identity issue include:

Sybil attack The Sybil attack takes place when a adversary initiates numerous malicious nodes on the system. These virual nodes join the system and start diseminating invalid files. If this can be done without any limit, the number of malicious nodes would be much higher than the number of real users, thus rendering the system useless. This can be regarded as a variant of denial of service attack.

This happens if 1) There is no cost to create a new ID; 2) No human interaction is required, thus allowing for automated peer creation. Centralized servers solve this problem by using "captcha"s or client puzzles, such as images that can be read or interpreted only by humans.Curiously enough, most papers on reputation management don't pay any attention to this problem and simply refer to these client-puzzles as a method for solving this. It is not clear who shall challenge a new node with a puzzle in a distributed environment. The bootstrap node[1]? Then malicious nodes will simply join through other malicious nodes who won't challenge them at all. All nodes? Seems impractical.

Another method might be to ensure (using cryptographic protocols) that the peer has performed a specific amount of computation, thus making creation of innumerable nodes computationally infeasibe. Unfortunately, this seems to suffer from the same problem.

[1]The node a new node connects through for the first time

Whitewashing Malicious peers change their identities when their reputation goes down. As identities are freely available, they just start all over with a clean slate. There are two school of thoughts on how to solve this problem. Some researchers think that the only solution to whitewashing is to have a centralized login server. The real world identity would be tied to one unique system ID. This would make changing identities difficult (if not impossible).

The other school of thought emphasizes on avoiding the centralized server, which creates a single point of failure. The problem is proposed to be solved by assigning new nodes low reputation ranking. This eliminates the incentives of changing identities. Researchers have proposed Initial reputation $p_0 = 0$.

Now this creates another big problem in that as new nodes start with a very low rating, they will not be chosen for downloads very often. Hence, it will be very difficult for them to build up a reputation. It seems that trying to balance these two issues will remain elusive in the light of the following impossiblity result

> No strategy ... can do substantially better than punishing all newcomers.[4]

A parallel with BitTorrent might be interesting. The reason why BitTorrent's mechanism of "Optimistic Unchoking"[2] works because of the immediacy of the domain. If a node is uploading now at a high rate, there is no reason not to upload to at in a high rate, regardless of its history. This is precisely not the situation in reputation management, where reputation is a asset built over time and hence has to related with the identity of the node, something BitTorrent can ignore as it only uses 20-second averages.

[2]Uploading to new nodes though they didn't upload anything

Chapter 7

Conclusion

7.1 Summary

Let us summrize what we have presented in this report:

- Peer-to-peer systems have emerged as a very popular network infrastructure of the time. Much research has gone into it and many working systems have been successfully deployed.
- Though its most popular use is as a file-sharing system, there is a good prospect of its being used for more seious purposes and evolving as a successful business model.
- With increased awareness about private and sensitive information, trust management is now an important issue regarding all networked infrastructures. This is also true for P2P, specially because of its open, anonymous nature. If P2P has to be successful as a business model, successfully managing trust is a prerequisite.
- Reputation management is the most widely studied form of trust management regarding P2P systems.
- There are two major ways to implement reputation management - limited reputation management and global reputation management. Limited reputation management is easier to implement and can work on any network structure. Simulation shows that it is robust upto 40% of peers being malicious.
- The global reputation scheme we studied is EigenTrust. It shows robust behavior even when 70% of the peers are malicious. However, it is difficult to implement, has a much larger overhead and can only work with distributed hash tables.

- Identity is a main problem in P2P systems. It will be fair to say that there has not been (and probably won't be) any scheme that is robust (to Sybial attack and whitewashing), allows complete anonymity but at the same time retains P2P's open nature (easy for newcomers to gain reputation).
- Probably the P2P community has to entertain the idea of having a centralized server for atleast indentity purposes (maybe multiple servers to avoid single point of failures). One idea maybe to allow the protocol work even if the server is down. The system will function even in face of server failure, but for that period of time it will be vulnerable to whitewashing. However, once the server comes up, the previous history is restored. Any misbehavior during the failure time will probably have to be forgotten.

7.2 Research Problems

We conclude by some open problems that have to be addressed. This list is not complete, a longer list can be found in [5].

- Examining the effect of weighting individual transactions in some way, rather than having a binary (satisfactory/unsatisfactory) value.
- Examining how long it takes for new nodes to build up a reputation in self managed ID systems.
- Having partial authenticity instead of binary authenticity.
- Study so far is quite intuitive, specially the limited reputation schemes. And work done so far depends mostly on simulation results to prove their claims. Providing a theoretical framework is a big challenge.

Bibliography

[1] Marti, S.; Garcia-Molina, H. Limited Reputation Sharing in P2P Systems, Stanford Database Group.

[2] Kamvar, Sepandar D., Schlosser, Mario T., Garcia-Molina, H. The EigenTrust Algorithm for Reputation Management in P2P Networks, Stanford Database Group.

[3] Marti, S.; Garcia-Molina, H. Identity Crisis: Anonymity vs. Reputation in P2P Systems, Third IEEE International Conference on Peer-to-Peer Computing.

[4] Friedman, E; Resnick, P. The Social Cost of Cheap Pseudonyms, Journal of Economics and Management Strategy 10, no. 2, pp 173-199. 1998.

[5] Daswani, N.; Golle, P.; Marti, S.; Garcia-Molina, H.; Boneh, D.. Evaluating Reputation Systems for Document Authenticity. Stanford Database Group.

[6] Adar, E. and Huberman, B. Free Riding on Gnutella. Technical report, Xerox PARC, 10 Aug. 2000.

[7] http://bt.etree.org/

[8] Cohen, B. Incentives build robustness in BitTorrent. http://www.bitconjurer.org/BitTorrent/bittorrentecon.pdf, May 2003.

[9] Faloutsos, M.; Faloutsos, P; Faloutsos, C. On power-law relationships of the internet topology. SIGCOMM, 1999, pp. 251-262

[10] The letter is featured on http://www.acenet.edu/washington/letters/.

[11] http://www.informatik.uni-bremen.de/ lechner/publications-ule/LechnerHummelEikemeierDASD2003.pdf

[12] Rheingold, H. The Virtual Community: Homesteading on the electronic frontier, Addison-Wesley. 1993.