TOWSON UNIVERSITY

COLLEGE OF GRADUATE STUDIES AND RESEARCH

THE NEW PEER-TO-PEER REPUTATION-BASED TRUST MODEL:

THE RESOURCE CHAIN MODEL

BY

SINJAE LEE

A DISSERTATION PRESENTED TO THE FACULTY OF
TOWSON UNIVERSITY
IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE
DOCTOR OF SCIENCE
IN APPLIED INFORMATION TECHNOLOGY

MAY 2008

TOWSON UNIVERSITY
TOWSON, MARYLAND 21252

TOWSON UNIVERSITY
COLLEGE OF GRADUATE STUDIES AND RESEARCH

DISSERTATION APPROVAL PAGE

This is to certify that the dissertation prepared by Sinjae Lee, entitled "The New Peer-to-peer Reputation-based Trust Model: The Resource Chain Model," has been approved by this committee as satisfactory completion of the requirement for the degree of Doctor of Science in Applied Information Technology.

_____	_____
Committee Chair, Dr. Yanggon Kim	Date
_____	_____
Committee Member, Dr. Ramesh K. Karne	Date
_____	_____
Committee Member, Dr. Alexander L. Wijesinha	Date
_____	_____
Committee Member, Dr. Yeong-Tae Song	Date
_____	_____
Committee Member, Dr. Robert J. Hammell II	Date
_____	_____
Dean, College of Graduate Studies and Research Dr. Chao Lu	Date

ACKNOWLEDGEMENTS

There have been a number of people without whom this dissertation could not have been completed. My deepest gratitude goes out to Dr. Yanggon Kim. Yanggon has tolerated an incalculable number of questions from me and has weathered that storm beautifully. He has helped me to develop a deep interest in the research process and has mentored me well. I also would like to thank my committee members: Dr. Ramesh K. Karne, Dr. Alexander L. Wijesinha, Dr. Yeong-Tae Song, and Dr. Robert J. Hammell II for their flexibilities with me, their perceptive comments, and their ongoing efforts to help me be a better writer and researcher. I also would like to thank Dr. Chao Lu, dean of college of Graduate Studies and Research. I would like to express my undying gratitude to my father, Dr. Joon-Bang Lee, for his love, support and for providing me a model of what a scientist should be like. In addition, I wish to thank my wife, my sisters, my two daughters, and the rest of my family for their love and support. Finally, this dissertation is dedicated to the memory of my late mother:

ABSTRACT

THE NEW PEER-TO-PEER REPUTATION-BASED TRUST MODEL: THE RESOURCE CHAIN MODEL

SINJAE LEE

Trust-reputation controlling mechanism, as an active self-controlling method is especially useful to automatically record, analyze and even adjust peers' reputation trust credibility among the different peers so that the system can adjust itself according to the credibility changes. This kind of self-adjusting system is suitable for the anonymous, dynamic and variable P2P environment. Using the Resource Chain Model, the new P2P reputation-based trust model, we can find the best resource location both effectively and efficiently while maintaining the system security. The goal of this dissertation is to find whether the model can increase the successful download rate when the number of transactions and the number of malicious nodes are increasing. The study's results show that RCM improved the performance. Thus, this approach enables one to find a better solution to find the best resource chain in a P2P community.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

AF	Appearing Frequency
ANL	Abnormal Node List
DAB	Digital Audio Broadcasting
DHT	Distributed Hash Table
DoS	Denials of Service
DVB	Digital Video Broadcasting
ERCM	Enhanced Resource Chain Model
FC	Friend Cache
FT	Failure Table
GSM	Global Service for Mobile Communications
GUI	Graphical User Interface
ID	Identification
IF	Impact Factor
IP	Internet Protocol
LAN	Local Area Network
LRU	Least Recently Used
MAC	Media Access Control
OORS	Online Outpatient Reservation System
P2P	Peer-to-Peer
PC	Personal Computer
RCM	Resource Chain Model
TCP	Transmission Control Protocol
TF	Trust Factor
TGrps	Trust Groups
TTL	Time to Live
UDP	User Datagram Protocol

Chapter 1. Introduction

This chapter will cover the overview of this research, including problem statements and solutions. This chapter will start with a background of some P2P reputation-based trust models.

1.1 P2P Overview

A P2P network is different from a client and server network. To put it bluntly, a P2P network does not have a client or a server. The advantages of P2P networks are valuable externalities, lower cost of ownership, and anonymity/privacy. Conversely, there are many malicious peers and attackers in the P2P communities because a P2P network is open and anonymous by nature. The attackers can easily spread the malicious content in a P2P community. As is well known, computer security has placed more attention and developed mechanism to increase the P2P security like encryption, sandboxing, reputation, and firewall.

Among those technologies, reputation mechanism as an active method is especially useful to automatically record, analyze and adjust peers' reputation trust histories among the different peers, and this method is suitable for the anonymous and dynamic P2P environment. Therefore, a reputation-based trust mechanism will secure a reputation system enough.

In particular, P2P is a class of systems and application that employ distributed resources to perform a critical function in a decentralized manner. The resources are computing power, data, network bandwidth and presence. In addition, the critical

function should be to distribute computing, to share data/content, and to provide communication/collaboration platform service.

Additionally, P2P is a technology that can communicate one to one between PCs without a server whose application area is very broad. After the appearance of Napster, people realized their computers have more powerful abilities than just doing jobs, which are simple browsing, downloading, and sending emails. Therefore, the meaning of desktop PCs changed to server instead of client. The benefits of a P2P approach are improving scalability by avoiding dependency on centralized points, eliminating the need for costly infrastructure by enabling direct communication among clients, and enabling resource aggregation.

The purposes of P2P systems are cost sharing/reduction, resource aggregation (improved performance), interoperability, improved scalability/reliability, increased autonomy, anonymity/privacy, dynamism, enabling ad-hoc communication, and collaboration.

The characteristics of P2P system are as follows:

- The public ownership aspect resource is used in a discrete form and is situated at the end of a peer.
- In a peer set, each peer uses resources that offer spreads to other peers who know the target resources: audio/video data, application, computing power, connectivity and presence information etc.
- Peers linked mutually to a network can scatter into the world.
- In a P2P network, each node can supply both client and server functionality. It is related to the sharing of distributed resources and services.

- Decentralization is a main characteristic of a P2P network.

There are many advantages of P2P networks. First, there are no additional cost about hardware and software. Next, a system can easily be set up. In addition, a network administrator and a user can control shared resources directly. The last two advantages are that there is no dependency on another computer for jobs and the system is inexpensive to set up. Despite many advantages, a P2P network has a weakness, lack of security. This dissertation will introduce a new approach on how to manage and control the security issues efficiently by using a reputation model.

Because searching for data is easy in a P2P system that promotes self-determination in a distributed system without centralization, the biggest problem is how to control and manage. Currently, there are two access ways, unstructured and structured P2P. There are two parts of P2P studies. One is an unstructured P2P and the other is a structured P2P.

The unstructured P2P searches resources and peers using search query flooding by a server or a neighbor peer as a structure; there is no relationship between resources and peers. The centralized model of P2P networking's early model managed a peer's IP address that possess resources be given by the server. This model was known by Napster; its use was limited by copyright of shared file [22].

By the alternative in reply, a pure P2P model was a way to search a query to do flooding by a contiguity peer without a center device until resources were found. However, problems that signal traffic by flooding produces a model of excess. In the proposed hybrid P2P, model to supplement this system, several super peer structures are composed to do a search query by flooding or efficiently spreading out the signal traffic.

The second part of a P2P is a structured P2P. This part consists of Distributed Hash Table (DHT). DHT offers distributed indexing [24] because a data search is available by a maximum number of search times O (logN) in this technique that increases the number of peer voluntarily without affecting the search efficiency (N is number of peer).

In addition, the quantity of necessary information of a structured P2P is less than P2P of an existing server base for searching at each node. Communication overhead for searching a structured P2P is less than a P2P method of flooding base. However, while complicated query was possible using attributed price of various data in unstructured P2P, there is a shortcoming because that query is simplified by a search that spends specification key by that use DHT in a structured P2P[24].

There are many advantages and disadvantages of a P2P system. If a problem of existent internet service happens to a server, all users' services are discontinued. However, if a specification server of P2P dies interrupted service does not happen to all users. Existing Internet information exchange contributes through only a server with a peer who wants to communicate to its neighbor directly. Users achieved completion that established their own server by these inconvenience. A peer of P2P system can exchange information justly with its neighbor.

A quality control problem exists in a distributed system with the burden of the maintenance of the program, the stability of the system operation and the believability problem. The problem occurs when users must have and act as a discrete peer. The server's function is excluded gradually; the performance of the system can slow down. The problem of security is advanced greatly. Otherwise, the danger continues that act as

kind of spy's role which peer program with bug collects information in user's hard disk. Businesspersons need authority that emphasizes customers' security in this side.

Decentralization includes algorithms, data, meta-data, etc. The advantages of P2P are valuable externalities, lower costs of ownership, cost sharing, and anonymity/privacy. However, problems in P2P are security and accountability. The core concept of P2P is about sharing resources between peers, decentralizing and leveraging vast amounts of resources. P2P is the opposite of the client/server model [20].

1.2 P2P Computing

File sharing is one of the most successful systems in P2P. Its features are large; multimedia's inherently large content files are available from multiple resources; anonymity is available to protect publisher and reader, and manageability for better performance. Today, bandwidth consumption, search, and security are issues in a file sharing system. Various types of collaboration from instant messaging to chat to online games are available. However, finding the location of peers is still a challenge by using a centralized server for peer location and out-of-band system to identify peers.

Another P2P computing system is Seti@home, which searches for extraterrestrial intelligence. Its background searches through massive amounts of radio telescope data to look for signals and to build huge virtual computer by using idle cycles on an internet computer that runs computation as part of a screen saver [12]. In addition, its features are fault resilience, since clients can stop at anytime, use check pointing every 10 minutes and horizontal scalability, but vertical scalability can still be a bottleneck. Another type is Gnutella, one of the distributed technologies that will reconstruct the Internet [12].

Decentralized technologies provide many desirable characteristics and Gnutella can improve such technologies.

Freenet is a P2P file-sharing system and the goal of Freenet is to make the system completely anonymous [12].For example, Freenet can make requesters and hosts unknown; individual nodes do not know what files they are hosting. Moreover, Freenet is completely decentralized and a pure P2P. Freenet must know of a node in the network to join; then nodes set aside a certain amount of disk space for hosting. When the disk space is full, Freenet uses the LRU strategy [40].

There are two different worlds in Gnutella [41] [42]. Gnutella is an alternative to Napster. Gnutella is not software like Microsoft Word. In addition, Gnutella is a protocol for communication and P2P file-sharing protocol over TCP [23]. It is not very scalable and requires a lot of bandwidth. In addition, it uses flooded requests algorithm. The Flooded request method can generate a lot of traffic because the number of possible responses increases exponentially with each hop. Ultrapeers is one of the solutions [27]. Nodes that keep routing tables of content and leafs nodes send keywords and metadata to Ultrapeers. One Ultrapeer can serve up to 100 leaf nodes.

Another system is JXTA. There are three layers in JXTA [11]. The first layer is the JXTA core layer, whose functions are discovery, transport, and creation of peers/peer groups. It is associated with primitive security. The following layer is JXTA service layer. The main purposes of this layer are searching, indexing, directory, storage, and file sharing. It has protocol translation, authentication, and PKI service. JXTA application layer is the top layer. Instant messaging, document/resource sharing, P2P email system, and auction systems are the characteristics of this top layer. In the future of P2P

algorithms, the world will become decentralized; P2P algorithm will have to overcome scalability, anonymity and connectivity problems. These P2P applications are most likely to succeed in the future, and P2P platforms will adopt JXTA widely.

There are three different algorithms [11] in P2P: centralized directory model, flooded request, and document routing. First, the centralized directory model, which uses an index in which all peers published information about the content they offer for sharing. The most popular example is Napster. Peers send a search to a central server with an index; then the server looks up the node with the desired file and gives the location to peers. Finally, the transfer proceeds without further server intervention.

Flooded request is the next algorithm. It is a pure P2P model and broadcasts to all nodes. All peers are connected to some other nodes, and then file requests are sent to all known nodes. If the node has the file, it returns it or if it does not, it sends requests to all nodes that are known. This algorithm continues until the file is found or time runs and stops.

The last algorithm is document routing, which is very efficient for large global communications and is the most popular P2P model. It has two problems. One is the difficulty of implementation and the other is islanding problems. The examples of this algorithm are Chord, Tapestry, CAN and Pastry [21]. The document routing model is efficient because it uses hash ID according to peer and file information. However, there are two disadvantages. It is hard to implement because ID is hard to find known before requests. The other is the islanding problem. It means sub-communities do not link to each other.

In P2P communities, the open and anonymous nature is a great threat to the security of the P2P community. The open and anonymous nature of a P2P network makes it an ideal place for attackers to spread malicious content. There are some security mechanisms to fight these attacks:

- Multi-key encryption: a public key, multiple private keys, asymmetric encryption mechanism
- Sandboxing: protect peer from malicious code and protect code from malicious peer
- Digital rights management: watermarking
- Reputation and accountability
- Firewalls

1.3 Related Work

Trust is a peer's belief about the capabilities of another peer. The capabilities are reliability and honesty, based on the peer's own direct experiences. On the other hand, the capabilities about reputation are honesty and reliability based on recommendations received from other peers [9]. The capabilities show risks, solutions, and trust principles if a person downloads software by using P2P environments. In addition, there are five frameworks for reputation-based trust [43]. These are trustworthiness, feedback, opinion, source, and destination of trust evaluation.

In a P2P system, one frequently meets unknown agents. The agents have to be trusted to avoid damages, which are sharing files containing viruses and redirecting queries. Besides trust management covers metrics for evaluation, data collection and

algorithms for the computing of trust. It is important to focus the first risk of the trust issues when downloading software. Reputation Systems are users who can access resources from other peers. They grow their opinions about credibility of those peers; then these opinions are collected and shared through certain mechanisms.

There are seven different models in reputation-based trust models:

- Complaint-only MODEL
- Feedback and credibility-based trust MODEL
- PeerTrust: A model which considers more factors
- EigenRep: Global Repuatation Value considered
- XREP protocol: Using Vote Polling
- EigenTrust
- Router Reputations

Firstly, begin with complaint-only model. There are three sub-problems: trust model, data storage management, and trust computation. The trust model is based on binary trust (either trustworthy or not). This problem can be solved using analysis of statistics collected or experience gained. Data management is storing the distributed data and the example is user P-Grid [17].

There is a simple example of a P-Grid, which shows each peer holding part of the overall tree [4]. Every participating peer's position is determined by its path, that is, the binary bit string representing the subset of the tree's overall information that the peer is responsible for Trust computation: the generalizing is

$$T(p) = | \{c(p, q) \mid q \in P\}| \times | \{c(q, p)| \ q \in P\}|.$$

It means high value of T(p) show that p is not trustworthy.

Advantages of the model are to make good use of experience especially the complaints, a reasonable way to store trust data using the P-Grid method in a distributed system, and generating a result, which can used to decide the further action. On the other hand, the big problem is that if p is honest, q is cheating, c (p, q) and c (q, p) both rise and another peer r cannot tell which one cheats in the transaction.

The only way the author gives out is to check another peer s to see if s trusts p or q to decide. This is not enough if q and s both cheat. Additionally, P-Grid's updating among peers is inefficient. Only complaint is considered, but there are many other factors to affect the P2P community.

Secondly, what should be remembered is feedback and credibility-based trust model. This is a simple model mostly based on the trust feedback and the credibility factors of the community to decide the later action. The transaction vector is the most important thing in this model. In general, the transaction history is stored in trust vectors, and there is an integer with each trust vector to show how many numbers are significant bits in this trust vector. After each action, the most significant bit is replaced by the latest result, and the history bits move to the right of the vector. It is enough to calculate trust rating and distrust ratings according to the vector's bits.

In the trust query process, a threshold number is given to show the number of peers needed to be considered and the number stands for the number of most trusted responses. It is possible to query less than the THRESHOLD number of peers using (1) where "c_i" is the credibility and "t_i" is the trust rating.

$$\frac{\sum_{i=1}^{k} c_i t_i}{k} \qquad (1)$$

There are two steps in this model [13]. The first step is after a successful download and the other is to calculate the trust and distrust rating. There is a figure which shows that advantages of this model are using this trust vector can let a dishonest dealing be hard to erase its unacceptable history and can let it have a better trust rating if he did a reasonable amount of community service and enhance security to avoid some security problems.

Enhanced security are denial of service protection: an extra round is added to the protocol to avoid malicious file downloads: using hash of the file and it is quite likely to divide the file into segments sized in the 100KB-1MB and each segment is assigned a hash then check the hash to avoid malicious files.

The big problem the author considers more is about the peers who give better trust responses. The author thought that it is reasonable because some low-trust responses may discredit a reliable resource, later limit the number of responses, and finally be a performance bottleneck. This model does not consider other factors, which may affect the trust decision especially like community environment factors. Lastly, the trust and credibility numbers are all stored in itself, and they may be changed by malicious users.

Thirdly, PeerTrust is a dynamic P2P trust model. There are five important factors in PeerTrust. The feedback a peer obtains from other peers, the feedback scope, such as the total number of transactions that a peer has with other peers, the credibility factor for the feedback, the transaction context factor for discriminating mission-critical transactions from less or no critical transaction, and the community context factor for addressing community-related characteristics and vulnerabilities [15].

Similarly, in an ecommerce community, many factors are considered [16]. In the first place, feedback is defined in terms of the amount of satisfaction. Most existing reputation based systems use this factor, but it is flawed if only using this information. The next is the number of transactions. This factor is important for the ecommerce community.

The third is the credibility of feedback. The transaction context factor is the next. In this ecommerce community, some malicious users may increase their trust rating by doing many successful small transactions and cheat in big dealings. In the final place only, the community context factor is to adapt the model to different situations.

The basic metric of PeerTrust is:

$$T(u) = \frac{\sum_{i=1}^{I(u)} S(u, i) \times Cr(p(u, i))}{I(u)}$$

This metric computes the trust value of a peer by an average of the credible amount of satisfaction the peer receives for each transaction performed during the given period.

Adding temporal adaptability is:

$$T(u) = \alpha \times \frac{\sum_{i=1}^{I(u)} S(u, i) \times Cr(p(u, i))}{I(u)} + \beta \times \frac{\sum_{i=1}^{Ih(u)} s(u, i) \times Cr(p(u, i))}{Ih(u)}$$

By assigning a proper weight, the history of the peer can be taken into account but with a lower weight than the recent history. Providing incentives to rate is:

$$T(u) = \alpha \times \frac{\sum_{i=1}^{I(u)} S(u, i) \times Cr(p(u, i))}{I(u)} + \beta \times \frac{F(u)}{I(u)}$$

The community context factor can be defined as a ratio of total number of feedbacks the peer gives to others during the given time period.

Accordingly, the general metric of PeerTrust is:

$$T(u) = \alpha \times \frac{\sum_{i=1}^{I(u)} S(u,i) \times Cr(p(u,i)) \times TF(u,i)}{I(u)} + \beta \times CF(u)$$

where I(u): Number of transactions performed by peer u, p(u,i): Other peers participating in u's I(u) actions, S(u,i): Satisfaction of u in its i th transaction, Cr(p(u,i)): Credibility of p(u,i), TF(u,i): Transaction context factor, and CF(u): Community context factor [16]. The first part of the metric is the average amount of credible satisfaction a peer receives for each transaction. The second part adjusts the first part by an increase or decrease of the trust value based on a community-specific characteristics and situations.

Advantages of this model are to consider many factors which may affect the model's similarity of reality; the general model can be easily adopted by different communities by changing some factor numbers; and easy to implement the main algorithm. On the other hand, there are disadvantages of this model: it is hard to define the many factors, which need statistics to show the relationship between model and a real P2P world. It does not provide a method on how to store the trust peers in distributed community; and to use the trust value, it is necessary to careful consider the threshold trust number, which is hard to get with this model.

EigenRep normalizes local reputation values [10]. As an example is:

$$C_{ij} = \frac{\max(s_{ij}, 0)}{\sum_j \max(S_{ij}, 0)}$$

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

- sat (i, j): number of satisfactory transactions of i with j
- unsat (i, j): number of unsatisfactory transactions of i with j
- The author defines a normalized local reputation value Cij

There are advantages of this model: it is more effective. Because it is based on local reputation and later, get a global reputation value and the use of mathematical matrix convergence to achieve the goal value, which may be implemented using recursive or using cache. On the contrary, the disadvantages of this model are that it is easy to divide the whole community into sub-communities, which do not communicate with each other if one malicious user exists in a certain community and this model does not consider much the global reputation computing algorithm complexity.

XREP protocol considers the resource reputation and peer reputation like peers have trust value, different resource also has different reputation, which will increase the system credibility [7]. The local reputation analyzing process is similar to other models. There are two significant processes, which are to begin with the vote process and to compare flooding requesting with other models.

The characteristics of the vote process are that the peer asks the other peers both the resources and the peers who have the resources. The pool request also includes a public key PKpoll, with which the poll responses will need to be encrypted, and upon receiving the request, each peer checks its experience repositories and return the message encrypted with the key PKpoll.

The second process is the vote evaluation. The peer uses decryption to detect tampered votes and discards them. Then, the peer randomly chooses a set of voters in the cluster and directly contacts each voter with a TrueVote message to confirm the credibility of the vote. Next, with the vote result, the peer chooses the best one and sets up a connection.

This model has three advantages. In the first place, it uses voting from many peers instead of personal analysis of the reputation value in order to increase the credibility of the best one. In the second place, with the request encryption, the peer can distinguish the malicious replies. Only in the final place, it recognizes cliques of dummy or controlled voters by clustering the voters' IP addresses.

However, there are three disadvantages. Firstly, encryption decreases the P2P network efficiency and if the peer wants to check voters' credibility, additional work needs to be done. Secondly, how to set up a reasonable relationship between voting requests and the credibility since it is also important to consider a voter's credibility and its reporting value which is complex and hard to implement. Lastly, clusters of malicious users may be hard to check out and it is possible that this voting will be controlled by a group of users who combine to lowly rate their competitors.

In contrast to global history schemes like EigenTrust, this model uses only limited or no information sharing between nodes [18]. Additionally, the author argues that with less information than global trust information, this model can also give out a reasonable trust value for a peer. The first procedure is Select-best. Clearly, this procedure selects the response from the response node with the highest rating.

The method requires that a node maintain an ordered list of the most reputable nodes. For this reason, the author calls this list a Friend-cache (FC) of maximum size FC. One of the extraordinary advantages is to introduce the Friend-cache, which accelerate the querying and maintains the trustfulness. On the contrary, relying on FC may have problems because of the P2P's variability attribute. Thus, it is impossible to rely on one certain trustful peer, which plays as a trusted server for a peer.

EigenTrust is a reputation system, which decreases the number of downloads of inauthentic files. The reputation system is based on this concept of transitive trust [9]. The solution is to give each peer a trust value based on its previous behavior. There are three steps in the model. First, compute local trust value. Then normalize local trust value to avoid maliciousness. The final step is to aggregate local trust value.

It is possible to show how to get a new local trust value. It is possible to get the new local trust value using a matrix C. The matrix C includes the local trust vectors of all the peers in the system. For calculating the new trust values in the local vector of peer "i" based on requesting a particular neighbor "j," the matrix C is multiplied by the vector i.e. t(new)= c x t(old) [6].

Reputation for secure routing uses the reputation value [39]. It has many advantages and a disadvantage. The advantages are increasing in throughput and non co-operative nodes are ostracized. However, the disadvantage is that poor nodes are penalized. The solution available is to use resource availability information along with the reputation value. Achieved equilibrium in traffic management means that good nodes receive more traffic, become overloaded, drop some packets and decrease their reputation, and source nodes use second rank nodes the system equilibrium is established.

1.4 Problem Statements

Current P2P applications can be classified into one of the following three categories: file sharing, distributed processing, and instant messaging. Since file sharing is the most common application for P2P networks, the focus of this research will be on P2P applications for file exchange [5]. Many researchers have dedicated a lot of effort

towards this reputation-based system area and some of them have made reliable theoretical models.

One simple model is mostly based on the trust feedback and the credibility factor of the community to decide the later action. The transaction histories are stored in trust vectors, and there is an integer with each trust vector to show how many numbers are significant bits in this trust vector. After each action, the most significant bit was replaced by the latest result, and the history bits moved to the right of the vector. It is enough to trust rating and distrust ratings according to the vector's bits.

Using the trust query process, which is based on the trust feedback and the credibility factor, there is a possibility of decision by the later action, calculate trust rating, and distrust rating by the responses of the responders. The transaction histories are stored in trust vectors; we assigned an integer with each trust vector to show how many are significant bits in this trust vector. After each action, the most significant bit was replaced by the latest result, and the history bits move to the right of the vector.

It is important to calculate trust rating and distrust rating according to the vector's bits. When someone wants to do a Trust Query Process, first a threshold number is given to show the number of peers. This number stands for the number of the most trusted responses. It is possible to query less than the threshold number of peers only if anyone wants to do any trust query.

The process is as follows:

$$\frac{\sum_{i=1}^{k} c_i \, t_i}{k}$$

By comparing the trust number with a standard number, this model helps peers make a positive or negative opinion of a possible destination peer. However, in this model, there

are unsolved problems. First, this model does not consider other factors, which may affect the trust decision. Second, it is difficult to decide a threshold credibility standard to judge even though trust thresholds are presented [36].

In a more complex model, the designer also includes some other factors to make the reputation system more powerful. Besides the feedback (S(u, i) to represent peer u's feedback of peer i) and credibility (Cr(p(u, i)): Credibility of p(u, i)), other transactions factors (TF(u, i): Transaction context factor) like transaction numbers (I(u)) and transaction scales are included to increase the transaction control. Also, the community context factors CF(u) are also included to make the model easily adaptable to different situations. The general metric of this model is

$$T(u) = \frac{\alpha \times \sum_{i=1}^{I(u)} S(u, i) \times (Cr(p(u, i)) \times TF(u, i))}{\sum_{i=1}^{I(u)} Cr(p(u, i)) \times TF(u, i)} + \beta \times CF(u)$$

new_credibility = old_credibility x (1 - Trans_Factor)

The first part of this general metric is used to collect all the transaction information and the second part is used to adjust the community effect on the final reputation result. This model successfully collects all the useful information of the P2P environment and seems very reasonable.

In the implementation of this model, the author uses cache technology to decrease the number of calculations of trust requests by putting a trust cache in each peers' local vector. However, too many transaction factors are hard to collect and this complex computation will certainly delay the processing.

One big problem is that such models concentrate on the relationship between the node and its direct neighbors. Usually when a peer wants to find a resource, the peer will

get the resource path through many peers and peers' neighbors; such a resource path carries the most efficient information of the future resource searching. Thus, it is possible that this resource chain makes P2P communities more reliable. If the end of the chain provides reliable service, it will strengthen the whole resource chain. However, if there is an unsuccessful download, the whole chain will be weakened. A new P2P reputation-based management trust model concentrates on this reasoning.

1.5 Solutions

Even though the previous models successfully collect all the information, there is a problem with the transaction and the final reputation result. The problem is that this model concentrates only on the relationship between the node and its direct neighbors. If a peer wants to find a resource, the peer will have the resource path through many peers. The peers' neighbors try to find a good resource chain, which makes the resource successful. It is important to make good use of this resource chain by strengthening the whole resource chain. If the end of the chain provides good services or weakens the whole chain it becomes unsuccessful when downloading takes place. That is what the new P2P reputation-based trust model concentrates.

As peers enter a new P2P community, peers can get some network neighbors. Later peers can search from their neighbors and neighbors' neighbors to find the resource. Since such findings are sometimes long chains, it is hard to use knowledge to find the best chain from such candidate chains and finally try to connect the end of the best chain to get resources. It does not matter whether the chains' credibility is low and high. The importance is in making sure the starting peer choose the most reliable one from all

candidates. The amount of truthfulness is calculated using the credibility of the nodes in the P2P network.

When a peer enters a community, the peer has its own identities (for example IP addresses in an Internet or MAC address). When a peer enters the system, the peer can broadcast requests to enter the system and wait for other peers' connection replies. After the peer collects enough replies, the peer can choose neighbors from the peers who give out replies for the incoming requests. Such a choice can be accorded to IP address similarities, it is important to find neighbors with quite different IP addresses so that the peer may get neighbors from different groups.

Since such a choosing is only a start, it is likely there are malicious users among them. A peer can still update its neighbor table to add new neighbors and delete malicious nodes according to later transactions. It is important to discuss it in the update credibility part. After getting so many assigned neighbors, it is essential to add them into the neighbor table and give each neighbor a credibility number 0.5, which is the starting credibility. When a peer begins to ask neighbors questions about the resource and later begin transactions, the peer has to update this neighbor list.

One problem is how many neighbors for a node are good. Another problem is related to the search depth, that is, when a node does not know where to download information, the node will ask its neighbors. Its neighbors will have depth 1. Then if its neighbors do not know where to download and forward its requests to the neighbors' neighbors, these indirect neighbors will have depth 2. Therefore, the node can forward the request to depth 3, 4 …L.

The number of neighbors can be reasonable so that one node can also refer to its limited neighbors and collect their collected ideas without a lot of computing effort. In addition, the length of the resource chain is not very long to keep the search efficiency at the same time. It is inevitable that such a searching route length covers almost all the nodes in the P2P community.

Such two numbers have relationships with the scale of the community like this:

- N: Number of neighbors for each node
- L: Length of a resource chain
- SCALE: Scale of the P2P community.

Theoretically, it is presumed that each node can have N neighbors, so if length of the resource chain is L, there is a total number of nodes:

$$1+N+N^2+N^3+N^4+.....+N^{L-1}=\frac{N^{L-1}}{N-1}=T\_Scale$$

Therefore, SCALE should be no less than this number and cannot be too much bigger than this *T_Scale*. *T_Scale* < *SCALE* < *N x T_Scale* is one possibility for this problem. To solve this problem, it is necessary to get the scale of the community at the beginning using statistics or other methods to predict it. Then there will be a compromise of N and L using the relationship between the two numbers, and scale of the P2P community. Such a choice of N and L can be adjusted according to the running results.

1.6 Proposed Ideas to Approach a New Reputation-based Trust Model

The basic idea of a new model is like human nature. It is possible that someone can find food to eat for his or her survival. In the P2P network, a peer always tries to choose the suitable one among candidates to meet his or her needs. The peer has to choose one in

a limited time even though credibility of the peer is very low. Because if the peer does not choose the one, the peer will be too hungry (in network, a peer will wait too much time waiting for the resource).

A peer just makes sure the peer does his best effort and makes good use of his neighbors' opinions to find the best one who can meet his needs within his neighbors. Through questions and responses, a peer can get many survival resource chains from his neighbors and neighbors' neighbors and so on, which indicate the routes for him to find the goal. Peers do not mind if their credibility is low or high but only that they are choosing the most reliable one from their candidates.

Therefore, there is no such a threshold credibility number to indicate a connection or not. It is essential to connect a peer if he is the best one of a group. This is the basic need to survive in the P2P community. It should not to be denied to use reputation-based trust system to judge if it is necessary to make a connection between peers.

In a sharing environment, it is obvious that it is necessary to speed up for sharing by setting up multiple connections with many peers at one time. If some peers have a very good reputation and every peer wants to get objects from that peer, this peer will be forced to slow down because of network congestion.

When in the new reputation-based trust model, it might seem reasonable to suppose that peers debate such things when considering joining the new reputation-based trust model. One peer's requirement of the sharing of its resources and trying to get a balanced distribution among a certain P2P community is to get a fair-sharing system. Subsequently, peers can implement their needs by systematically analyzing factors included in their reputation representations.

P2P communities require the peer to install a program on their nodes that will work with other P2P nodes to implement the system. Since many applications can be built on a generic P2P substrate, it is interesting now the distribution of these codes support P2P applications. Users should not necessarily trust arbitrary programs, written by third parties, to run on their systems. For P2P systems, where applications can perform significant computations and consume vast amounts of disk storage, it would appear that general-purpose mobile code security architecture is necessary. However, there is a protracted way to go.

If there are different system levels of P2P computing, it is possible that there is this classification of levels in P2P computing [4]. The author presents architecture for trust management, which relies on all system, layers, namely network, storage and trust management, on P2P mechanisms. It is important to observe that in such architecture a mechanism implemented at a higher level in a P2P manner has always to take into account the properties, in particular the quality of service, of the mechanisms of the underlying layers. In a new reputation-based system, that which is important is a classification of such kinds of P2P computing and later such kinds of reputation in community or not needs consideration.

Chapter 2. The Resource Chain Model

This chapter describes a new reputation-based trust management model based on the resource chain model to prevent the spread of malicious content in the open community.

2.1 The Approach

The RCM is based on the idea of using a routing table to record the information about the credibility of nodes and their recommending nodes. Over the past few years, numerous studies on the reputation mechanism; however, there is no studies on the RCM. Peers that use the RCM can find the best and safest resource location efficiently and can decrease the number of malicious transactions in a network.

There are two difficulties in using the RCM. One is that the previous models only concentrated on the relationship between a node and its direct neighbors. The other is that so many transaction factors are difficult to collect and the complex computation required to do that would certainly slow down the processing. Clearly, peers in a P2P network should try their best to send packages to their destinations the same way that like UDP does in network transmission.

The main challenge is how to aggregate the local reputation ratings in a distributed environment without a centralized storage and management facility. For previous P2P algorithms, it was necessary to know the threats to the RCM before it was used. Denial-of-Service attacks disable or prevent the victim peer from using its network connection.

Every peer in a P2P network has to respond to a query from other peers. This requirement to respond can easily be exploited by malicious peers that collaborate by

repetitively sending matching queries that can eventually cause the network to respond continuously to malicious requests and finally to be unsuccessful.

Furthermore, there may be malicious users who send forged documents between users in a P2P community so it is necessary to check the reputation of the users by using RCM. Most users in a P2P community are willing to share the information that they have and to promote P2P development. Unfortunately, some users benefit from the P2P community and do not share their resources at all. The RCM also takes into consideration such “Free Rider” problems [29].

The RCM is based on several concepts. The purpose of setting up the RCM is to distinguish the malicious users from the good ones. Peers’ reputations serve as a benchmark for determining whether peers are malicious or good. Furthermore, a central server on which to store information about the reputation of peers is unavailable. The RCM relies on the P2P infrastructure for obtaining such information if it is not locally available at the querying peer. Hence, the topology of the RCM needs to follow that of the P2P community.

The RCM also stores transaction results into a local vector and forwards the information to querying machines if necessary. The transaction results that call feedback are the most important feature in RCM. However, there is a danger that a malicious peer may give negative opinions of some benign peers. It is possible that malicious peers can give a negative opinion to other peers intentionally because they are selfish or careless. Furthermore, there may even be users who change the data stored in their trust vector. Thus, the RCM should use additional control factors to correct or to adjust to the reputation result.

In addition, the RCM needs to have a credibility control factor to show the credibility of the feedback from peers. When the RCM sends out a request to determine whether a peer has previously acknowledged that it is trustworthy, the RCM asks for the opinions of those peers that have previously acknowledged that they are trustworthy. The RCM also gets feedback from other peers about the results. However, the RCM must then decide whether the opinions of the peers are trustworthy and then collect the useful information while discarding the untrustworthy information. Because the trustworthy information is disregarded such information may be forged opinions from acknowledgers.

The RCM uses credibility control during the filtering process to get opinions each with a different weight, from numerous peers. The result is that the overall opinion about the trustworthiness of the peer in question is more reliable than the overall opinion obtained from simply averaging the opinions of the other peers. The updating process for credibility is more difficult than that for feedback because a new positive feedback is generated and must be taken into account each time a successful transaction occurs. Nonetheless, if a peer gives a negative opinion as a part of positive feedback, then the credibility is stored as a negative opinion of the peers in the transaction, because the peers taking part in this transaction are being deceitful.

The RCM increases the credibility only when the status of a transaction is consistent with the opinions of the previously acknowledged trustworthy peers about the peer in question. For example, if a negative opinion of an unsuccessful transaction is given, the credibility is increased because the peers are telling the truth. This credibility control will help to make the RCM more reliable.

It is necessary to add some other community control factors to RCM because P2P communities vary greatly and because sometimes a model that can be adapted to a wide range of situations is needed. By changing these factors, the RCM can easily be adapted to different situations while considering different points. Such community control factors help the RCM to get a more reliable view of other peers in various environments.

The RCM should also give peers a decision about whether to connect. Therefore, it is necessary to assign some thresholds to indicate the minimum requirements for the trustworthiness of a peer or maximum threshold of untrustworthiness. A peer's trustworthiness can be determined by comparing the trust factor and threshold number [13] [16]. Such threshold numbers can be generated by considering all the useful P2P trust factors; they may be a result of a statistical analysis of history information.

The key point of the approach is to find the best destination from among numerous candidate destinations; however, it is not always guaranteed that such a best candidate is reliable. In a P2P network, the RCM always tries to choose the most suitable destination among numerous candidates from which to download information. The criterion for determining which node is the most reliable depends on the credibility of that node in the network. Figure 1 shows what happens when a peer enters a new P2P community.

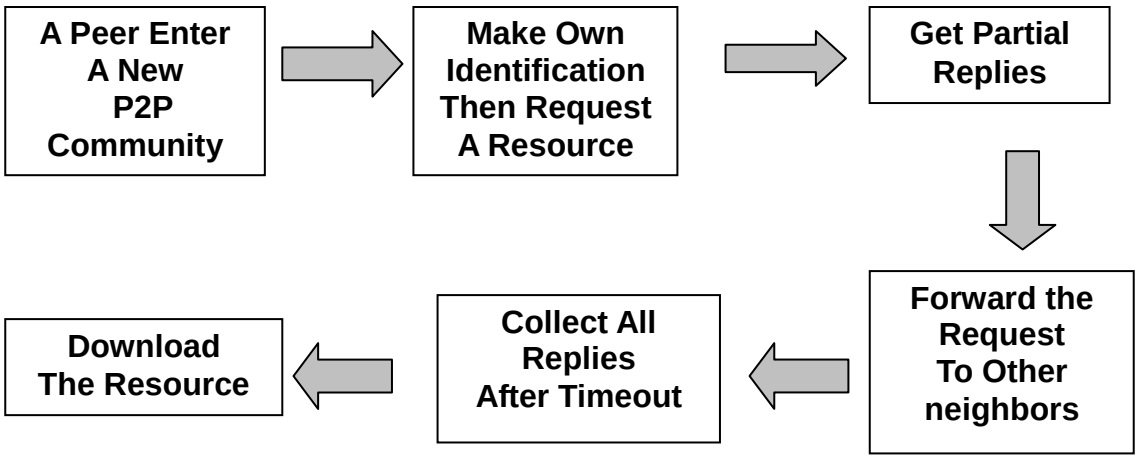


Figure 1: A Small Scenario

First, peers have their own identities, IP addresses on the Internet or MAC addresses. When a new peer enter the system, it can broadcast or multicast its requests to enter the system and wait for connection replies from the other peers in the system. After the new peer collects enough replies, it chooses neighbors from among the peers that replies to its outgoing requests. It chooses neighbors on the basis of the similarity of its IP address with what of its neighbors, that is, peers try to find neighbors with IP addresses quite different from its own so that peers may get neighbors from numerous groups. Since choosing neighbor peers is only the first step, there are likely numerous malicious peers among them. Peers can still update their neighbor table to add new neighbors and delete malicious nodes on the basis of later transactions.

Peers can then get some network neighbors. Later peers can search by asking their neighbors and their neighbors' neighbors to find the resource that they want. A search path includes the node that sends out the request and all the other nodes to which the

request is sent until the sending node gets the resource that is requested. Search paths are sometimes long chains, so it is essential to find the best chain from such candidate chains and to connect to the end of the best chain to get the request resources. Peers do not mind if their credibility is lower than some threshold credibility, even though this makes them seem less trustworthy. Peers should choose the most reliable path from all candidates. The amount of truthfulness is calculated using the credibility of the nodes in the P2P network. After the peers select one destination from which to download the resource, they can then download the file from that node.

The attempt to download the file from the requesting node to a node that claims it has the requested resource is successful. The RCM updates the credibility relationship from the starting node to the ending node by increasing the credibility between them. If the transaction is not successful, then the peers can decrease the credibility between the starting and ending nodes. After the transaction, the RCM rewords the successful transactions by increasing the credibility and vice versa. Such an updating is the most critical part in the RCM because it is efficient and self-adjusting.

2.2 Factors to Set Up

Feedback is given in terms of the amount of satisfaction a peer obtains through transactions with peers. Feedback is often a simple aggregations of the positive and negative feedback that peers have received for the transactions they have performed, so it and cannot accurately represent the trustworthiness of peers.

The credibility of the feedbacks is submitted by the peers. It is thus possible to determine whether a peer’s feedback is trustworthy based on the credibility. However,

there is a danger that malicious peers may form a group to promote the credibility of each other. Considering the opinions from more than one group is important in deciding whether feedback is acceptable. As with feedback, only using only positive and negative values to determine credibility is not reliable. Researchers have thus been developing many methods to make this factor more accurate.

The transaction control factor is another issue. Each instance of feedback is based on one transaction. The feedback is important; hence, it is necessary to pay attention to the transaction's control factors, like the frequency of a peer's transactions. For example, it is necessary to consider the number of transactions the peer has performed with other peers. This is because it is necessary to consider the situation in which a malicious peer may increase its total number of transactions to reduce the percentage of its malicious transactions.

Adaptive factors of different communities are also important. Different communities pay attention to different aspects of the RCM. A fine example of this is that, pop music communities only need to analyze the recently and frequently used pop music files to collect the feedbacks given in the last six months. Nevertheless, classical music communities need to pay attention to files of music made hundreds of years ago, and the feedbacks of years ago should also be collected. The author has introduced two factors: the transaction context factor and the community context factor [16]. With such adaptive factors, the RCM can easily be used in a different environment by only changing only the adaptive factor number.

2.3 Assumptions

It is necessary to make some assumptions to simplify the explanation of RCM [1] [2]. Each node in the system has eight neighbors, so the length of each search is 8 (Figure 2). Each node has a neighbor list to record information about a neighbor's nodes.

Assuming that the nodes are ordered by their credibility, the higher the node's credibility, the higher the position it holds. In addition, the following formula represents the relationship between nodes: for $0 < i < 9$, N_i = Neighbor (N_0, 1). This means that distance between nodes N_i and N_0 is one. It is assumed that at first, N_0 has the following neighbor list (Figure 3).

It is appropriate to use the above temp table shown in Figure 3 for each request to record all the temporary transaction results before the request is met. The temp table has a transaction ID for each node-like R-Chain [19]. Additionally, it is assumed that the maximum number of neighbors for a node is eight; however, this number can be adjusted in the real test.

2.4 How RCM Works

In this section, we shall discuss how the RCM works.

2.4.1 Request File

Figure 2 shows that node N_0 wants to know where it can download File X in the network. Because it only knows a limited number of neighbors, it sends the request to all his neighbors. "Request" means those that can tell peers where to download File X. Nodes N_1 through N_8 represent the neighbors of node N_0 through which it can directly ask for the resource location. Figure 3 shows the list of neighbors through which node N_0 can directly ask for the resource location.

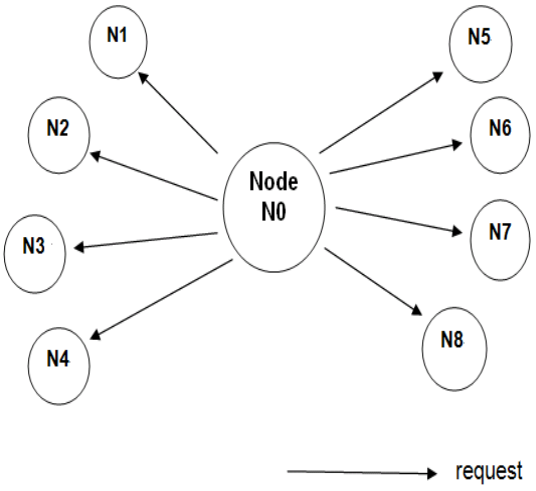


Figure 2: N_0 sends out request to neighbors

Neighbor List of Node N_0	
NODE	Credibility
N_1	Cr[1]
N_2	Cr[2]
N_3	Cr[3]
N_4	Cr[4]
N_5	Cr[5]
N_6	Cr[6]
N_7	Cr[7]
N_8	Cr[8]

Figure 3: List of Neighbors of Node N_0

2.4.2 Get Partial Replies

For those nodes that directly know where to download the File X, it is possible to establish one hypothesis. They will send a reply to N_0 without forwarding more requests. Here, there are new nodes that were previously unknown by node N_0 and N_1, N_2, N_3 and N_4 know where N_0 can download X. If N_i knows where to download X, the N_i will send a reply to tell N_0 where to download X. A temp table records the transaction information such as the nodes that N_i recommend and the information source they get from the nodes' neighbors.

- N_9 = Neighbor(N_1, 1)
- N_{10} = Neighbor(N_2, 1)
- N_{11} = Neighbor(N_3, 1)
- N_{12} = Neighbor(N_4, 1)

In the Figure 4, *Reply_N_i* means that if N_i knows where to download X, N_i sends replies to tell N where to download X. If N_i knows more than one node where node N can download X, N_i will tell N the most reliable one node. After that, a temp table (Figure 5) is necessary to record for each node the temporary credibility information of the nodes and that of their recommended nodes. All nodes that take part in the transaction use a temporary table to record the transaction information, such as the nodes they recommend and the information source.

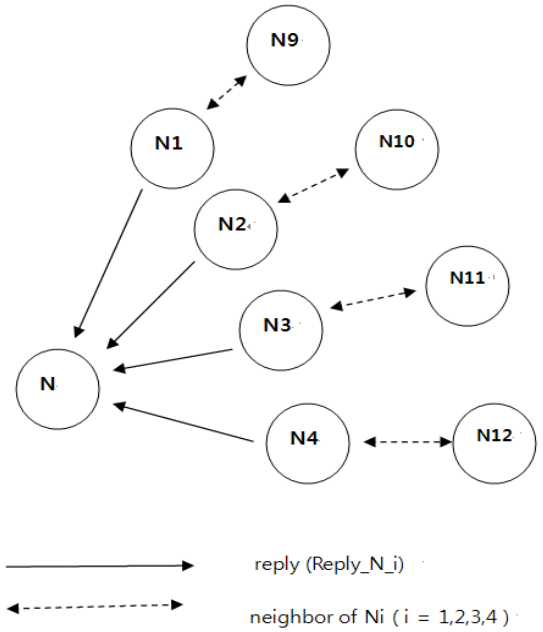


Figure 4: Node N gets some replies

Temp Table For Node N0

Neighbor ID	Neighbor Credibility	Recommend ID	Recommend Credibility
N1	Cr[0]	N9	Cre_0[0]
N2	Cr[1]	N10	Cr_1[0]
N3	Cr[2]	N11	Cr_2[0]
N4	Cr[3]	N12	Cr_3[0]

Temp Table For Node N1

Neighbor ID	Neighbor Credibility	Recommend ID	Recommend Credibility
null	null	N9	Cr_0[0]

Temp Table For Node N2

Neighbor ID	Neighbor Credibility	Recommend ID	Recommend Credibility
null	null	N10	Cr_1[0]

Temp Table For Node N3

Neighbor ID	Neighbor Credibility	Recommend ID	Recommend Credibility
null	null	N11	Cr_2[0]

Temp Table For Node N4

Neighbor ID	Neighbor Credibility	Recommend ID	Recommend Credibility
null	null	N12	Cre_[0]

Figure 5: Temp Tables for Node N0, N1, N2, N3 and N4

2.4.3 Forward Requests

Here, assume that N_5, N_6, N_7, and N_8 do not know where N can download File X, so they forward the request to their neighbors and introduce other neighbors to these four nodes:

- For 13 < i < 20, Ni = Neighbor(N5, 1);
- For 21 < i < 28, Ni = Neighbor(N6, 1);
- For 29 < i < 36, Ni = Neighbor(N7, 1);
- For 37 < i < 44, Ni = Neighbor(N8, 1);

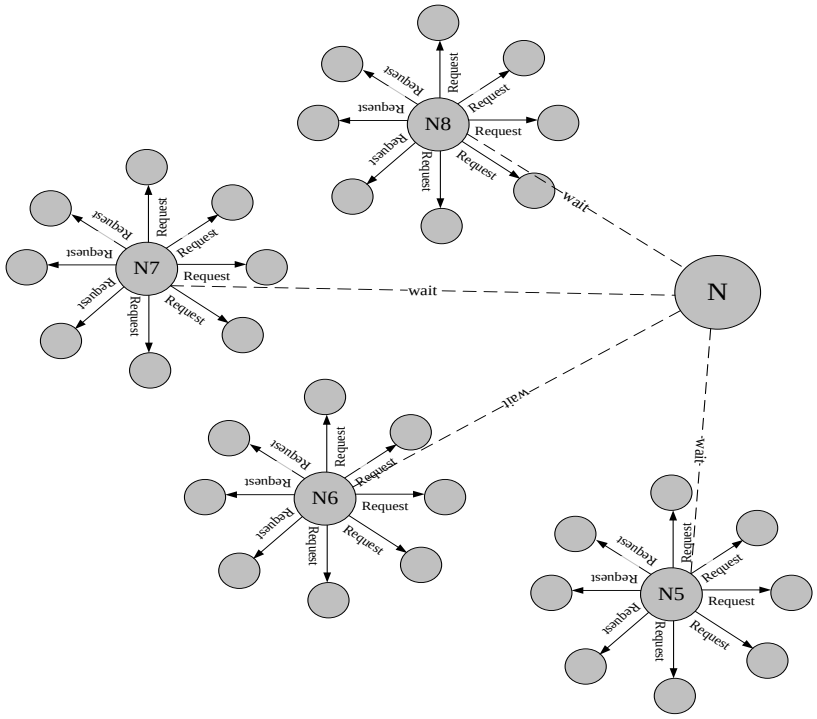


Figure 6: Some neighbors of Node N forward requests

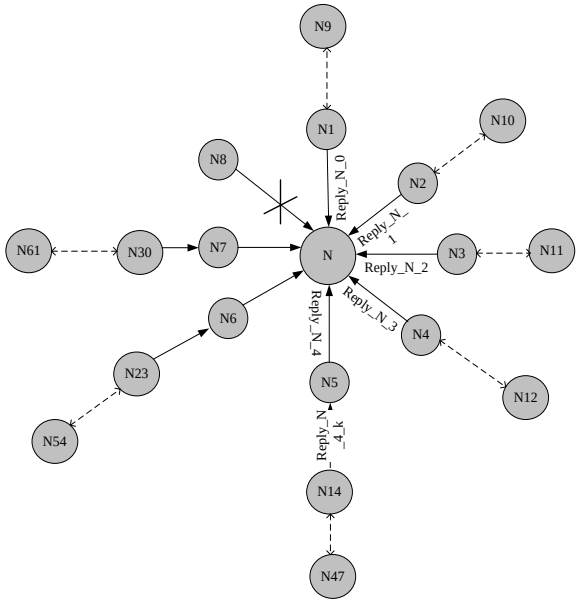


Figure 7: Node N gets all replies

Figure 6 shows that N forwards the request to its neighbors, which do not know where the needed resource is at that moment. Figure 7 shows that node N waits a while for replies from its neighbors. N stops waiting when a network timeout occurs, so after a while, the neighbor nodes will forward their findings. This can also happen if some neighbor nodes cannot tell N where to download. Above all, the RCM uses TTL.

2.4.4 Collecting All Replies After Timeout

There are four situations in which all replies are collected after a timeout:

- The node that sent requests can get numerous replies.
- After a timeout of the request, N will get information about paths to numerous nodes from which to download the file.

- Node N can choose *Neighbor_Credibility x Recommend_Credibility* from its temp table as the criterion for determining which neighbor node is the most reliable one from which to download the file.
- Some nodes may not provide any useful information in response to the request. For example, here, N8 here cannot get any useful information, so it did not give a reply (Figure 7).

Figure 8 shows the temp table for each node that has to be changed.

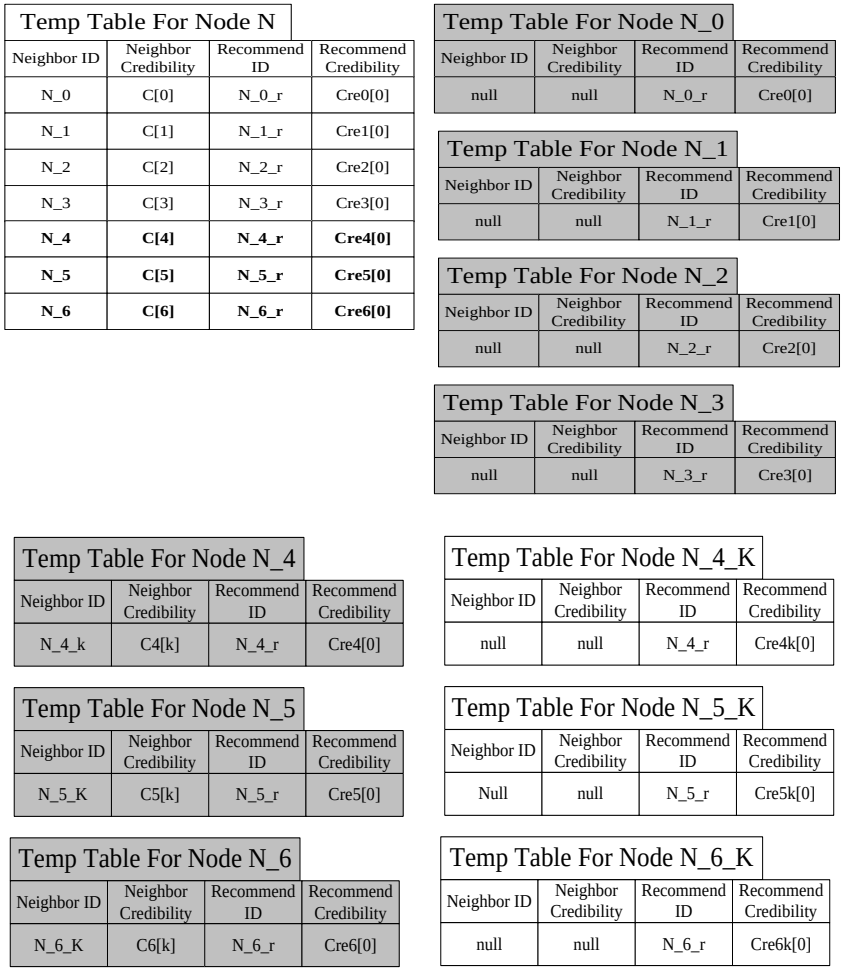


Figure 8: Temp Table for Node N, its neighbors and its neighbors' neighbors

2.4.5 Downloading the File

Node N chooses the node with the most credibility to download the file. If successful, return; if not successful, N chooses the next best path to try.

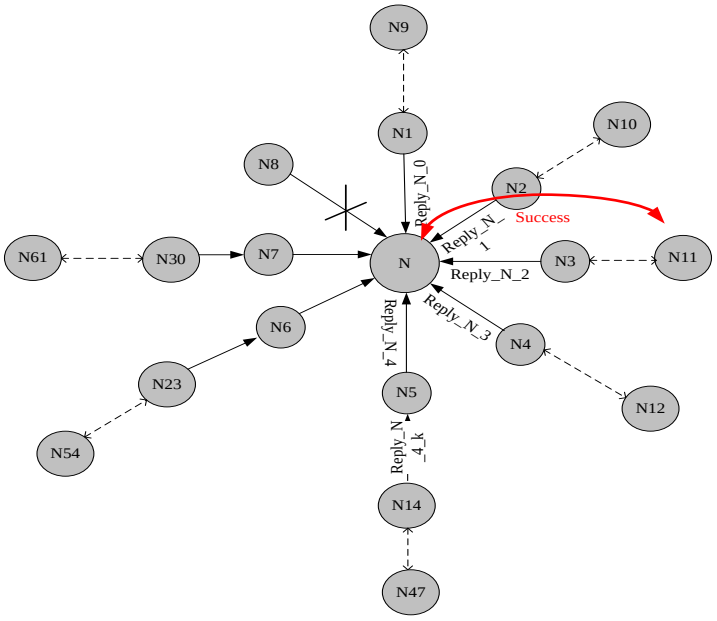


Figure 9: Transaction is successful for Node N

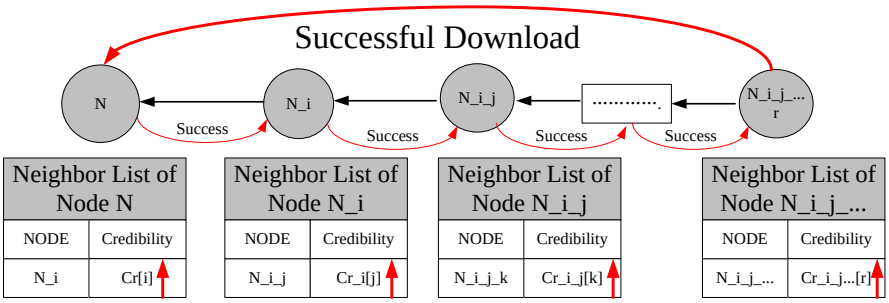


Figure 10: After successful transaction

It is possible that node *N_2_r* is the best candidate node from which node N can download File X. In Figure 9, N successfully downloaded the file from the best candidate node, N11. Node N then tells N3 this is a good recommendation and raises the credibility of N3 to thank N3. When N3 gets the information that the download was successful, it is glad that its recommendation is good, so it also raises the credibility of its neighbor, N11, which offers the resource for node N to appear.

It is possible to adjust the resource chain credibility after this update of their neighbors' credibility. The neighbor credibility of each node taking part in the transaction recommendation and downloading is updated after the transaction happens. In Figure 10, a neighborhood chain is strengthened after one successful transaction and the order of the nodes should be adjusted after one transaction on the basis of the nodes' credibility.

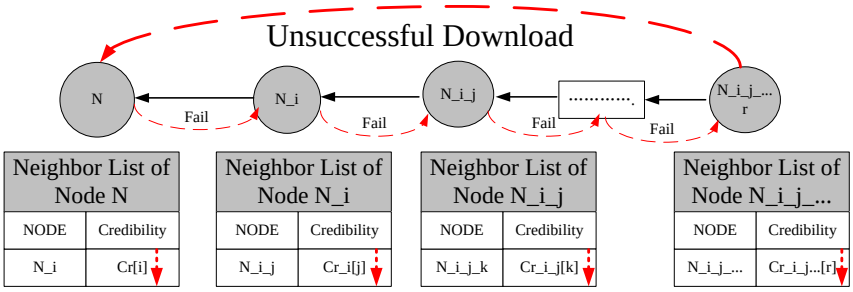


Figure 11: After unsuccessful transaction

The neighbor credibility of each node taking part in the transaction recommendation and downloading is updated after the transaction happens. It is then appropriate to adjust the resource chain credibility. Figure 11 shows the neighborhood

chain that is weakened after one unsuccessful transaction. For successful transactions, the whole resource chain is strengthened. However, for unsuccessful transactions, the whole resource chain is weakened.

2.5 Transaction Factors and Update of Credibility after Transactions

The most important factor to consider is the size of the file being downloaded. The worst case is when the file is very large, and some malicious user may offer to download 99% of the file. This would be a great waste of downloading bandwidth and time. The usual ways of preventing this situation from happening the following:

1. Adding file authentication information that is useful. Using other authentication methods can improve the file downloading credibility. An example of such authentication methods is the File Reputation System [14].
2. Dividing a huge file into parts, and downloading each part separately. It is thus appropriate to use a number *FILE_SIZE* to indicate the size of each part of the file. An important point to emphasize is the transaction factor. After the transaction, it is essential for a node to update the credibility for its neighbors.
3. Updating of credibility:

- The transaction factor is what is most important when using file-size information *Trans_Factor = log(file_size) / log(max_file_size).* It is necessary to stabilize this number. If the transaction is successful, *new_credibility = old_credibilty x (1 + Trans_Factor).* If the transaction is unsuccessful, then the equation for updating the credibility will be *new_credibility = old_credibility x (1 – Trans_Factor).* For

each node in the resource chain, it is essential to update this credibility on the basis of the same *Trans_Factor*.

- Log(*file_size*) is the transaction factor that can replace the file_size information. (There is no maximum file size in the calculation.) The running time usually measured by *Trans_Factor* = log (*file_size*); (*Trans_Factor* range: 1-N). The change factor is given by *Change_Factor* = *Trans_Factor* / (1 + *Trans_Factor*); (range: 0-1). It is necessary to stabilize this number again as in this Factor = Log(*Change_Factor* + 1); (range: 0-1). Therefore, when credibility increases, the result is given by *new_credibility = old_credibilty x (1 + Factor).* For decreasing credibility, the result is given by *new_credibility = old_credibility x (1 – Trans_Factor).*

The RCM has two stages. The first stage is collecting all the node information and selecting the appropriate node to download the file. The second stage is beginning the download of the resource from the node selected from among the candidates.

2.6 Tree Problems

One problem to consider for the RCM is finding the best path for each neighbor. It is usually possible to get more than one resource path for each neighbor, and such resource paths can be good candidates for use in downloading files. The most widely used and easiest tree to use is the binary tree, and using binary trees is appropriate in the RCM.

It is appropriate to concentrate first on the direction of constructing a resource tree when constructing it. The resource tree is assumed to start from a root because it is only possibly to record the two best neighbors or another neighbor that seems to be better than they are. It is usually incorrect to predict which one can provide the most useful path information. It is very likely that some reliable neighbors cannot provide paths and that some less reliable neighbors may offer some choices while nodes collect them because of their record in their parent nodes.

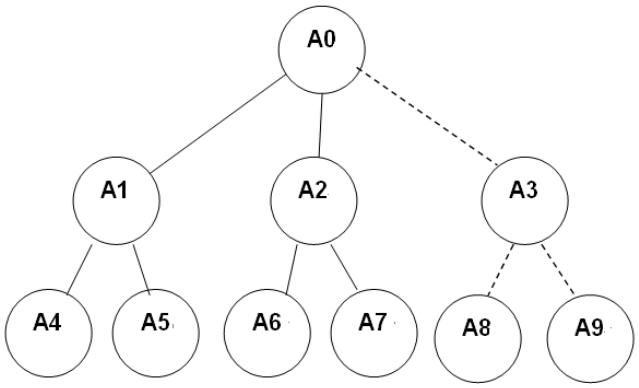


Figure 12: Tree Architecture

In Figure 12, for example, A1 and A2 seem reliable while A3 is not as reliable. Thus, at the first depth, it is a valid to record A1 and A2. However, only A3 can recommend the nodes necessary for downloading the resource. Thus, the only possible way to construct the tree is by starting from the bottom. Moreover, the root node has *NO_of_NEIGHBOR* children while all other nodes will have two children at most. It is clear that for a tree of depth N, the maximum number of leaf nodes the tree has is given

by $2^{N+1}-1$. In the RCM, only the leaf nodes are the resource owners. Now it is necessary to set up the resource tree.

Each node at depth N-1 can record as much as two nodes of information. Such depth N-1 nodes can record the two best candidates of all the resource candidates. Then the nodes will forward their information to nodes at depth N-2 to set up the tree. Because of the tree topology, each node at depth N-2 each can record at most the information of the four best leaf nodes. The problem here is four such leaf nodes may not exist as children nodes of two depth N-1 parent nodes; it is very likely each leaf may have a different parent. In that case, it is appropriate to record at most two depth N-1 nodes, which include the most number of best 4 path information regarding credibility.

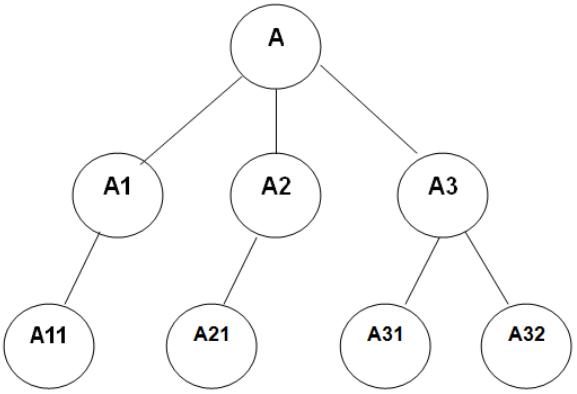


Figure 13: Best candidates of two-level architecture

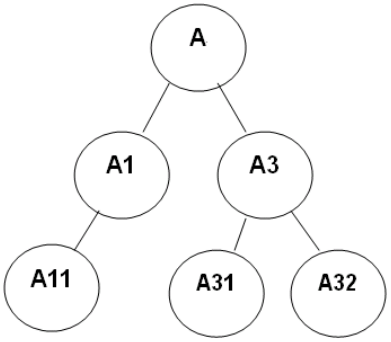


Figure 14: Peers' binary resource selection

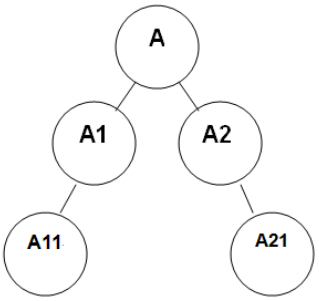


Figure 15: Another Peers' binary resource selection

In Figure 13, A is at depth N-2. Assuming such a resource tree, A11, A21, A31 and A32 are the best resource locations. If the order of each leaf node's credibility for node A is A31>A11>A21>A32, it is appropriate to set up A's binary resource tree like the one shown in Figure 14. However, if the order is A11>A21>A31>A32, it is possible to set up A's binary resource tree like the one shown in Figure 15.

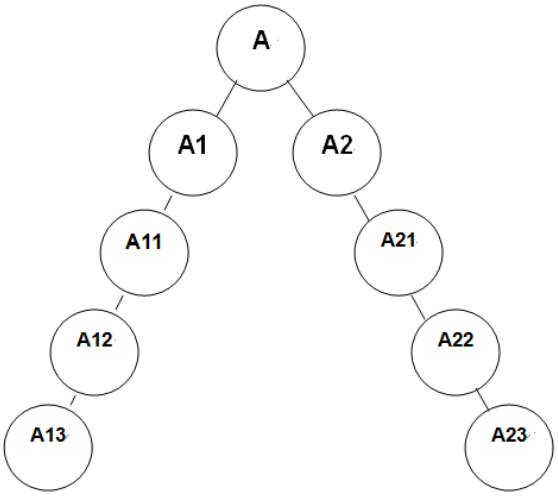


Figure 16: A theoretically extreme binary resource selection case

Each node at depth k (N-1>k>0) can record at most 2^{N-k} leaf nodes of information. It is important to restrict the number of children of each node to two. Usually, the root node has up to *NO_of_NEIGHBOR* children. Each child of the root node then, forms a binary tree. The resource location is recorded in the leaf nodes in the tree. Compared with the simple chain model, even in the worst case, as described in the following diagram, each child of the root node can record only two chains as shown in Figure 16.

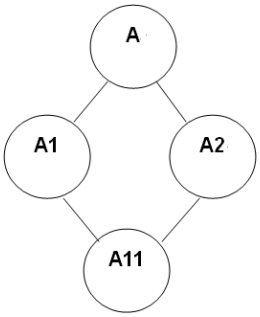


Figure 17: Graph problem

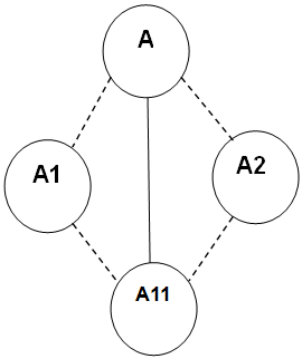


Figure 18: Break up of relationship in graph problem

The result is still twice as good as that obtained using the RCM because theoretically, each child will have more than two resource paths. In the chain model, only one path is recorded for each neighbor. There are two more problems with using the tree approach. The first problem occurs when two parent nodes recommend the same child node (Figure 17). The A11 node can be a leaf node or some intermediate node. The best solution is to break up the relationship between A, A1, and A2, and to set up a direct connection between A and A11 (Figure 18).

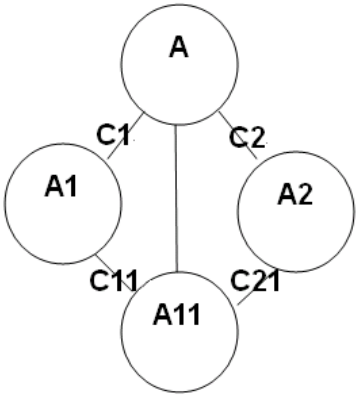


Figure 19: Simplification of 2-level credibility to 1-level

It is necessary to collect A1's and A2's opinions together, given that the credibility information is located here (Figure 19). Calculating the credibility of A11 for A is given by {C1 x C11 + C2 x C21 + 2 x (C1 x C2 x C11 x C21)} / 2. Such a calculation factors in both direct neighbors' opinions while enhancing the average a bit because of multiple recommendations.

The second problem is updating of the credibility of the neighbor nodes. As with the RCM, the difference in credibility depends on the first problem when both nodes recommend the same node, and it is necessary to interact with this node. In that case, it is appropriate to let A1 and A2 improve or to decrease C11 and C21 more than usual in order to break up the graph interaction. The graph problem is the updating of credibility: For the direct neighbors, there is nothing to do; only for the node to the combining child node is possible to change the credibility to be bigger than usual and to greatly decrease the credibility chain.

2.7 Identifying the Malicious Nodes

This section describes the analysis and identification of malicious nodes by using frequency identification.

2.7.1 Some Observations on the Behavior of Malicious Peers

The behavior of malicious peers is one of the most researched areas of the P2P world. We presented the new reputation-based trust model developed in previous research to prevent the spread of malicious content in the open community by using a resource

chain model. In spite of the previous studies, the malicious behavior of peers in the RCM has never been examined.

The purpose of this section is to identify a malicious node in the RCM. While the previous model has global observation, this one has local observation. This approach means that the behavior of a malicious node changes the security level in the RCM. Thus, what needs to be emphasized is more detail about the local view of the RCM. Moreover, this research aims to analyze the behavior of malicious peers on P2P networks.

There are numerous kinds of threat scenarios [9]. One kind is malicious individuals or malicious peers that always provide inauthentic files [35]. Another kind is a malicious collective. Two malicious peers that know each other give each other good opinions and give other peers bad opinions. A third kind is a camouflaged collective. This is a tricky kind because a malicious peer sometimes provides authentic files to deceive good peers by giving them good opinions. A fourth kind is malicious spies are those peers of the collective that always give good files, but that always give good feedback to malicious peers.

Furthermore, numerous types of cheating are possible in the real P2P world [28]. A peer can exaggerate credibility. This means the peer intends to have a high credibility even though it actually has a low credibility. Another type of cheating is conspiracy. To put it precisely, a malicious peer can evade detection by using malicious neighbors. A third type is blame transfer. A malicious peer may try to blame a good node for hiding other malicious peers' misbehaviors. A fourth type of cheating is when a malicious peer can delete some information from other malicious peers' lists of interested peers by hiding consistency or by sending malicious information.

Numerous malicious actions can happen in the real P2P network [30]. First, attacking actions are possible. Eclipse [31] and resource-consuming attacks, distributing corrupt data, and attracting peers without serving them are representative of attacking actions. Abnormal behaviors, such as frequent joining/leaving and free riding, are also possible. According to "Resilient Trust Management for Web Service Integration [32]," the threat model presents two malicious behaviors - providing inauthentic web services and distributing dishonest opinions. On one hand, inauthentic web services can be harmful by propagating viruses and on the other hand, distributing dishonest opinion may cause a peer to choose less trustworthy providers.

In addition, malicious behaviors are correlated to attack models [33]. There are four different types of attacks, which are membership, DoS, and omission attacks, and ultimately, forgery [34], that can explain malicious behaviors. Malicious peers provide good service and try to join Trust Groups (TGrps). After joining a TGrp, malicious peers always provide bad services.

Concerning the numerous possible ways of cheatings [28], it is necessary to study the behaviors of the malicious nodes. The first phase [1] [2] is related to a simple malicious behavior. On one hand, this research aims at studying the behaviors of malicious nodes and at finding the differences between the first and the second phases; on the other hand, the purpose of this analysis is to justify why it is necessary to use the second phase.

It is indispensable to identify malicious peers because the original update information is not sufficient for the RCM to identify that malicious information. More approaches on real-world P2P for increasing the rate of successful downloads deserve

careful attention. Although the RCM has been achieved some progress in network security, the limitations of the RCM oversimplify the malicious nodes' activities.

It is appropriate to consider the malicious nodes' of various harmful activities in order to make the RCM more realistic and closer to the real world P2P network. The activities of the RCM's malicious nodes will always be opposite to those of normal nodes. For instance, although a malicious node does not have the resource, the node can pretend to have it. The importance of concentrating on malicious nodes' behaviors cannot be overemphasized in further improving RCM security.

Recent studies have shown that the resource chain, the best chain selection, the credibility update for the whole chain, and the neighbor list are maintained in the previous works [1] [2]. It is important to focus on collective historical transaction data and on the statistical analysis of data about the failure transaction table. It is also important to identify malicious node instantaneously. Having decided that the RCM has many advantages, the only thing left to discuss is its limitations due to the behaviors of malicious nodes. Recently, the RCM has attracted a considerable amount of attention as more realistic network model that closely approximates a real-world P2P network so it is important to identify and reveal such malicious nodes.

The limitations of the model due to malicious nodes need to be examined in detail. It is clear that malicious nodes will always do the opposite of normal nodes, as stated in the previous papers [1] [2]. Consider this situation for example, even a malicious node does not have resources; however, the node pretends that it does. Such a node is simple to identify only if it always does the opposite of normal nodes; however, sometimes the

node does the same thing as normal nodes. Therefore, it is very difficult to identify whether the node is malicious.

An important concept in the RCM is that data tables are stored in each node and will play a key role for data exchange. In the previous model [1] [2], the forwarding table information never changes so it is important to consider malicious node behaviors because they are common in a real-world P2P network.

2.7.2 Approach

The key point of our previous approach is that we try our best to find the best destination but it is not possible to guarantee that the candidate nodes are reliable. Consider a small scenario. All peers have their own unique identify. After the peers download resources and the transaction is successful, then the credibility of the nodes between the start and destination nodes is increased. However, if the transaction is unsuccessful, then the credibility of the nodes between the start and destination nodes is decreased.

Although the RCM is the local reputation model, the RCM can be considered as a part of a global reputation model, like that of broadcast or multicast models. Because it is a local reputation model, the RCM is a pure P2P model. As a result, the second phase also has local reputation mechanisms.

Another factor can be introduced to identify malicious nodes. That factor is the honesty factor; H. H represents the probability that malicious nodes are telling the truth. At first, there is no difference among different malicious nodes' H values. H is more like a statistical number whose value can be set at first, and, if necessary, gradually be

updated as more data is analyzed. The thing behind this number is the assumption that malicious nodes will not always do the opposite; sometimes, they will hide deeper.

It is most important to consider the malicious nodes in the second phase. One would not find useful data for malicious node analysis after a successful transaction. Although malicious nodes may have taken part in the transaction, they behaved positively; thus, no useful data is available for malicious data analysis. As a result, the RCM can use the data of failed transactions.

A failed transaction sometimes occurs. When it does besides the updating in the RCM, the credibility-updating process needs to inform the starting node (the node that sent out the initial request), so the node broadcasts or multicasts the transaction chain to everyone in the community or to whichever nodes have taken part in the transaction.

2.7.3 Failed Transaction Chain and Frequency Identification

The importance of the failed transaction chain in the second phase cannot be overemphasized. Peers broadcast or multicast chain information to everyone. In short, this is simplified chain information, which means that somewhere between starting node N0 and N33, for example, there must be a malicious node. Consequently, all nodes in the RCM that get a table like the one shown in Figure 20 keep it for future analysis.

Chain ID	*Failure Route*			
1	**N0**	**N3**	**N12**	**N33**
2	**N1**	**N4**	**N5**	
3	**N7**	**N8**	**N9**	**N11**
4	**N5**	**N7**	**N21**	

Figure 20: Table of Failed Transaction Chain

The method of identifying malicious nodes is called frequency identification. This could be the easiest and most efficient way to reduce the amount of risk in a P2P network. Those nodes that appear most frequently in peers' failure table would rarely be believed by other nodes Figure 20 shows a simple failure data table. Node N12 is possibly not reliable.

After some time, every node might have a table of failed transactions. For example, find the nodes that appear frequently in failure tables then compare their appearance frequency (AF) to H. If AF is greater than H, then those nodes are treated as malicious nodes. Here is a numerical example of this concept. Assuming that H = 0.5 and AF (12) = 4/5 = 0.8 > 0.5, node 12 must be malicious (Figure 21).

Failure ID	Nodes List (Size varied)								
1	0	3	7	8	1	12	21	34	67
2	2	7	32	43	18	43			
3	4	3	12	54	16	55			
4	6	2	12	87	43	77	15		
5	7	9	12	87	23	92	23		

Figure 21: Failure Table

The frequency could be misleading, however, as some of the nodes in between only act as forwarders, so it is essential to find more reliable method of discovering malicious nodes. Data mining is necessary for defining some own-secure pattern or sub-route, hence, partially preventing the frequency effect of good nodes.

Malicious nodes also send out this kind of failure transaction chain. They are actually totally faked. Because the chain is faked, the nodes, which are in the faked failure transaction chain, will know it is faked. Consequently, they directly know that that node is malicious. They will send out a correction message to warn others that the node is malicious. In this way, if a malicious node cheats and sends out faked information, it will help others identify it.

2.7.4 Faked Failure Scenario

Malicious node N0 broadcasts out a failure table (Figure 22). Thus, nodes N3, N4, N6, and N7 directly know that N0 is malicious because they did not take part in this transaction and because they can check their transaction histories. Good nodes broadcast back malicious node identification messages (Figure 23). Figure 23 shows the message from N3. Thus, the left nodes, which do not know N0 is malicious at first, like N1. The node will get fake chains from N0, also malicious identification messages from N3, N4, N5, N6, and N7. Therefore, the node will believe several other nodes, not only one.

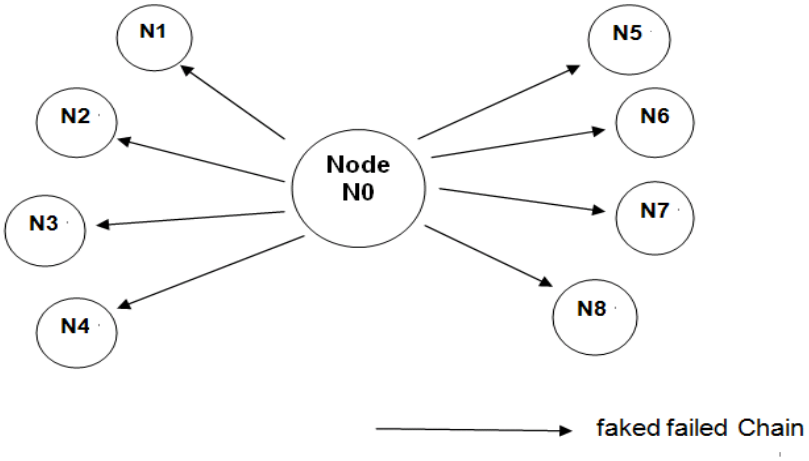


Figure 22: N0 Broadcasts Out Faked Failure Information

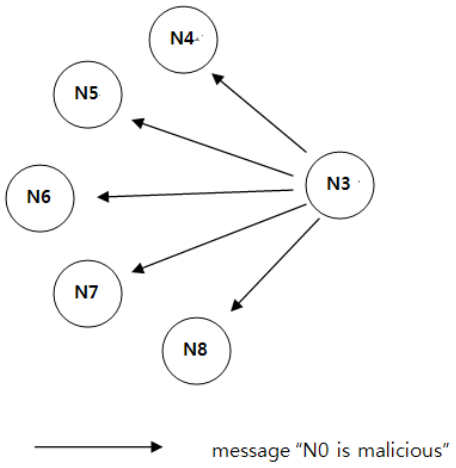


Figure 23: Good Nodes Broadcast Back

2.7.5 Solutions

By analyzing the possible risks and threats in P2P networks, we can actually get the following three most popular threat models. Our improved resource chain model can be powerful enough to handle these possible risks to make our P2P network community secure.

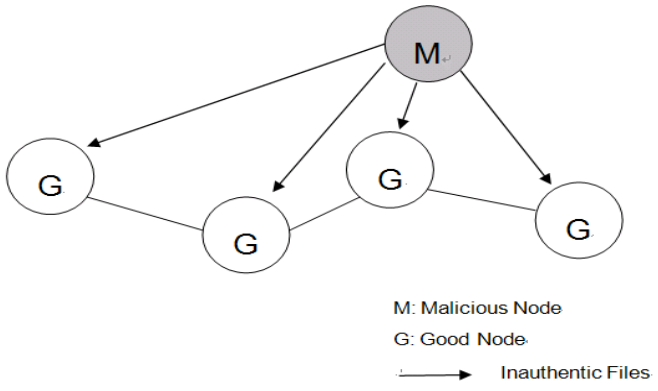


Figure 24: Malicious Individuals

The first type of threat model is actually called “malicious individuals” (Figure 24). The definition of “malicious individual” comes from [9] and means a kind of peer that always provides inauthentic files. By applying this kind of threat to the RCM, we can say that in this kind of threat, a malicious node is always doing the opposite of normal nodes (those nodes that always provide authentic resources or information).

For example, given that there is a resource request by a node, which means the node is looking for the resource. Assume that another node gets this information and that it is a malicious node. As a result, although it has the resource, it will refuse the request. However, this action will not be easy because of the improved RCM.

In the improved RCM, a method called “forwarding requests” is used when a node does not have the resource. Specifically, assume that a malicious node has the request. The node will respond that it does not have the file, even if it does, and it will return the request by using “forwarding requests.” This means that he will do exactly the opposite of normal nodes. In the first phase, whenever a transaction is finished, it is necessary to update the credibility based on the result of that transaction. This means that if it has a successful download, the RCM will give the node high credibility.

It is essential to update peers’ neighbor lists depending on the credibility differences [1] [2]. Thus, malicious nodes will be given a lower credibility by nodes directly or indirectly connected with them. Given that more transactions happen, the malicious nodes become increasingly isolated from the “normal node group” because other nodes update the malicious nodes’ credibility time after time. This method is helpful for maintaining the system’s credibility. By dynamically updating credibility, the method slowly removes malicious nodes from our interacting group.

If we say that the method of dynamically updating is somehow a passive or an indirect method of prevent the first threat, only after transactions, we would try to do the updating and the result of this updating would help prevent this kind of threat to some degree. However, by doing that, we can only get relationships between malicious nodes; we can never truly say we identify the actual malicious nodes. Identifying the actual malicious nodes is more powerful than identifying the relationships between malicious nodes, and it helps to directly increase the security of the network system.

In the second phase, another method can be used to identify the malicious nodes. The method is called "Failure Transaction Collecting and Analyzing." It is based on the data collected from failed transactions. The method works like this: whenever there is a failure, the node acknowledges the failure and multicasts the failure data to its neighbors and to its neighbors' neighbors. Each node holds a kind of failure transaction table.

Whenever a transaction fails, assume that there is/are some kind of malicious node(s) in the route between the starting node and the destination node. For the first threat, the malicious node(s) will always does/do the opposite of the other nodes, thus it is necessary to look at the failure data collected. It is possible to get a list with each node's failure count by counting the number of times that a certain node is listed in the failure table. Malicious nodes have large failure counts.

Assume that the network has about 10% malicious nodes. The malicious nodes can be identified by picking the worst 10% of the nodes listed in the sorted failure count table. Peers will never connect with such nodes again. After getting the lists of the worst 10% nodes, it is necessary for peers to see the neighbor lists and to monitor all the possible transactions. These kinds of "malicious nodes" always perform these tasks poorly; thus,

their behaviors will result in a bad reputation. The best way to confirm such nodes as malicious nodes is to check their credibility in the neighbor lists or through transactions.

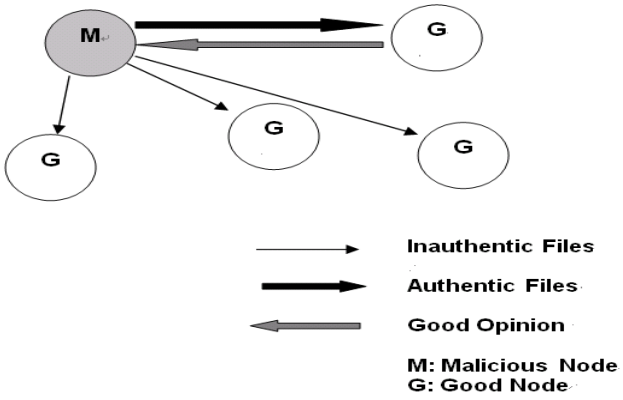


Figure 25: Camouflaged Nodes

The second type of known threat is camouflaged nodes (Figure 25) [9]. This kind of malicious nodes is "smarter" than the first kind of nodes, so they are actually more difficult to recognize. The reason that camouflaged nodes are smarter is that these kinds of nodes can partially provide authentic files to get good opinions; these kinds of nodes are the same as "malicious nodes." However, even this kind of security threat is difficult to prevent in real-world P2P networks.

First, the credibility updating strategy is fair to every node that has taken part in transactions for a certain camouflaged node. Given that transaction T[0] with node list {N[0], N[5], N[x]} has finished successfully, if N[x] is doing well, it is given a good reputation from both N[0] and N[5]. However, another transaction T[1], that has the transaction list {N[5], N[0], N[x]} was unsuccessful. In that case, N[x] was a camouflaged node. Thus N[x] is given bad credibility value from both N[0] and N[5].

After T[0] succeeded and T[1] failed, the credibility of N[x] for N[0] and N[5] remains almost the same considering the symmetry of updating either positively or negatively. Assume that N[x] is doing more good things than bad things in a particular node's opinion. That node still slightly increases its credibility rate for N[x].

Our updating strategy is useful because if nodes behave badly, they will have the danger of getting a low credibility rate and would possibly be isolated in the future on the basis of their bad reputation. If these kinds of camouflaged nodes could perform more good transactions, they would get the same reputation or an even better one than normal nodes on the basis of the transaction quantity and quality.

However, the second method "Failure Transaction Collecting and Analyzing" will also work here because it is possible to focus on the failed transactions, so no matter how smart these nodes are, whenever they do something badly or not, it is essential to log that event.

The difference between camouflaged nodes and malicious nodes is that the camouflaged nodes partially do things well. The camouflaged nodes can have a result in better reputation than "pure" malicious nodes who always give wrong information. It is necessary to introduce another factor called the "trust factor." The trust factor is a statistical indicator that indicates the probability of the camouflaged node behaving in a truthful manner.

Normal nodes have a trust factor of 100%, while malicious nodes have a trust factor of 0% because they never send true information. Camouflaged nodes can have "trust factors" between 0 and 100% and are typically about 40%. In 60% of these cases, the nodes will do well, having 3 successful transactions for every two false ones. As a

result, these nodes get a credibility promotion of 3-2=1 for every five transactions, so these nodes gain a better reputation as time goes on. It is essential to determine whether some nodes are camouflaged by assuming that their trust factor is 40% in the neighbor list.

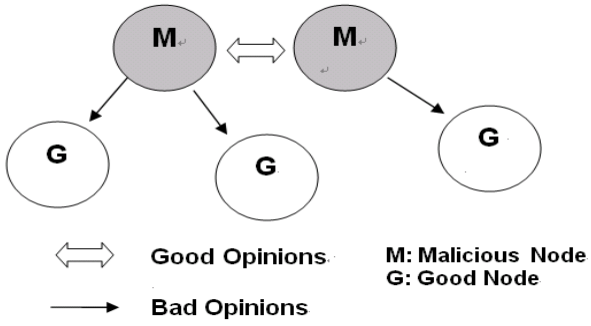


Figure 26: Malicious/Camouflaged Collective

The third type of threat is the malicious/camouflaged collective which is the trickiest of all threats (Figure 26). In this case, malicious and camouflaged nodes combine to form a group, and they give each other good opinions and occasionally give bad opinions or lie during transactions when interacting with other nodes.

The strategy for updating credibility would be the perfect first step in dealing with this kind of threat. It is important to focus on the "chain" of the nodes in the RCM so that the starting node can find a destination node to get a requested resource. Given that two or more nodes are in the same chain and that they are badly performing transactions, they both will get poor reputations; and no matter how much they increase each other's reputations, they will be unable to regain their trustworthiness.

It is also necessary to analyze the failure table data in order to consider the possibility of malicious/camouflaged collectives. If a node is at the end of a failure table, that node will have the most impact because it is the final node. If it happens to be malicious or camouflaged, each node from their neighbor lists will also be placed in that failure table, which will give normal nodes a partially negative reputation in the future. In that case, nodes at the end of a failure table should be given more attention and given more severe penalties when the nodes are caught cheating. Similarly, those nodes that come before the end of a failure table are given comparatively less severe penalties.

The next node, which is two nodes away from the destination node, would be given less severe penalties. An easy way to make use of this changing "impact factor" is to use the distance of the node in question away from the starting node. When we do the simulation, we are quite likely to identify the malicious or camouflaged nodes.

2.7.6 Summary

The two main components of the reputation system are capability and behavior [8]. It is therefore reasonable to analyze various peers′ malicious behaviors. Moreover, recent studies have shown that there are many possible malicious peers′ behaviors in real-world P2P networks. While the previous model has global observations, this one has local observations such as various malicious behaviors. We must be concerned about the behaviors of malicious nodes if we are to increase the level of security in the RCM. In the previous work, the RCM assumes that malicious nodes never tell the truth. However, in this research, malicious nodes will partially tell the truth. The trust factor affects the percentage of their truthfulness. In addition, if initially a download is unsuccessful, the

whole resource chain is weakened. Conversely, peers forward failure data to their neighbors′ neighbors in this phase. Most importantly, we can identify malicious nodes by analyzing the data in the failure table.

2.8 The Enhanced Resource Chain Model

In the previous model [1] [2], malicious nodes always do the opposite of normal nodes. We can use the two advantages of the RCM, namely, dynamic resource-chain-credibility updating and dynamic neighborhood maintenance. In addition, the previous simulation [2] has data, which successfully supported the RCM. It is important to dynamically maintain a sufficiently credible neighbor list and to increase the total rate of successfully downloaded resources when the number of transactions is increased in order to enhance the security of the P2P community.

There are various kinds of potential risks and threats in P2P networks. The RCM has some limitations in identifying malicious nodes, and it has difficulty in preventing other threats. In the enhanced resource chain model, it is necessary to focus on the enhancement of security, which takes into consideration more information about possible risks and threats and the solutions to them. It is also possible to identify the malicious nodes in the P2P network by using the histories of peers′ behaviors.

2.8.1 Risks and Threats

As described in the previous section, there are many kinds of malicious behaviors. The most popular ones are the following three kinds of behaviors:

Case 1: Malicious Nodes

These nodes are the same as the nodes in the RCM. The nodes do the opposite of normal nodes.

Case 2: Camouflaged Nodes

These nodes behave smarter than the malicious nodes by partially telling the truth to confuse other normal nodes.

Case 3: Malicious/Camouflaged Collective

These nodes are the most dangerous. They not only behave maliciously, but also give good feedback to each other and give bad feedback to normal nodes.

The RCM has partially solved the problem of malicious nodes by dynamically updating neighbor lists. Hence, the malicious nodes in the RCM are isolated. It is very important to focus on enhancing the RCM with respect to the Case 2 and Case 3 threat groups.

2.8.2 Identification of Abnormal Nodes

The RCM was focused on increasing network security by maintaining reliable neighbor lists and by dynamically updating peers. However, the RCM only finished the work of isolating the abnormal nodes from the reliable nodes. It was previously impossible to identify the abnormal nodes, which behave maliciously in the RCM. In the enhanced resource chain model, it is necessary to use a new feature called ″Failure History Analysis,″ which can overcome the limitation of the RCM.

It is essential to define some new terms before implementing the enhanced RCM. These terms are used to show the characteristics of different groups of nodes and to

represent the new functionalities of each group of nodes.

■ Trust Factor (TF)

To simplify the analysis and grouping of all the potential malicious nodes in the RCM, we must use a TF value for each node in P2P networks. The TF is a percentage between 0% and 100%. It indicates the probability that a certain node is going to tell the truth. It is important to categorize the different groups of nodes in the RCM on the basis of their TF.

- Normal Nodes: TF=100%; Normal nodes always tell the truth.
- Malicious Nodes: TF=0%; Malicious nodes never tell the truth.
- Camouflaged Nodes: TF=x%; Camouflaged nodes have varied TFs.

  X is between 0 and 1, which means that the camouflaged nodes sometimes tell the truth.
- Malicious/Camouflaged Collective; each node has a different TF.

■ Failure Table (FT)

Assume that there are failed transactions in P2P networks. After the failure, there are updated credibility chains and updated neighbor lists. Assume that nodes record and multicast the failure chain data to neighbors and that each node in the RCM records the failure data into the FT for analysis, which identifies abnormal nodes. The FT is a table structure in each node, which records all the failure transactions. In the RCM, the data in the FT can be analyzed to identify abnormal nodes.

- **Abnormal Node List (ANL)**

The ANL is similar to the neighbor list, which stores the identities of nodes. However, the ANL only stores the identities abnormal nodes that are possible future threats.

- **Impact Factor (IF)**

The Impact Factor is a new value in the enhanced RCM that enhances the updating of the credibility chain. The ″IF″ is a relative indicator that depends on the position of each node in the RCM. If a node is close to the starting node, then that node has a high ″IF.″

2.8.3 Updating Credibility

After a certain number of transactions, nodes will update their neighbor′s credibility lists on basis of transaction results. Figure 27 shows the updating scenario after a certain number of transactions.

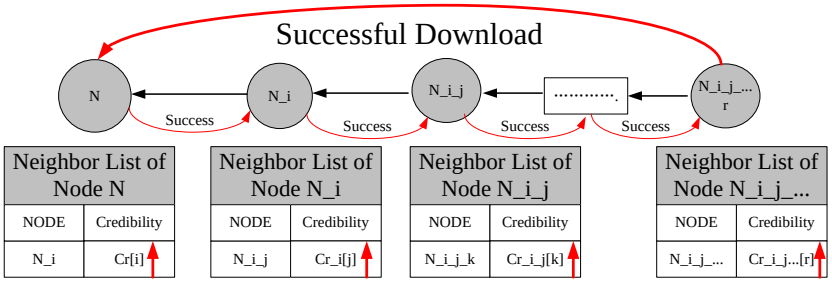


Figure 27: Updating Scenario

Figure 27 shows that each node in the chain gets the “successful” response from node N, the starting node. Therefore, each node in the chain increases its credibility for its directly connected neighbor. The following formulas are used to update credibility in the enhanced RCM.

Credibility is updated based on formulas F1 and F2 after a successful transaction.

- **F1: CRnew = CRold x (1+Award_Factor)**

  CRnew is the new credibility after updating and CRold is the old credibility. Award_Factor is the bonus factor that indicates how much we should increase the credibility. After we introduce the impact factor (IF), the formula will be replaced by this enhanced formula, which takes into account the location effect in the RCM.

- **F2: CRnew = CRold x (1 + AwardFactor x FLoc_Eff (Location_of_Chain))**

  FLoc_Eff is a function for calculating the IF on the basis of the location of each node in the RCM.

Credibility is updated based on formulas F3 and F4 after a failed transaction.

- **F3: CRnew = CRold x (1 - Penalty_Factor)**

  Penalty_Factor is a weight that indicates how much the enhanced RCM should decrease the credibility of a node.

- **F4: CRnew = CRold x (1 - Penalty_Factor x FLoc_Eff(Location_of_Chain))**

2.8.4 Analysis of the Failure Table

After a node has a failure table (Figure 28), the failure data in it can be used for identifying abnormal nodes. There are many steps in the identification process. The first step is to refine the abnormal candidate nodes′ group. This begins with the frequency

statistics of each node in the failure table.

Failure ID	*Failure Chain*							
1	**N4**	**N0**	**N1**	**N2**	**N6**			
2	**N7**	**N9**	**N0**	**N1**	**N2**			
3	**N8**	**N0**	**N1**	**N2**	**N14**	**N16**		
4	**N6**	**N7**	**N8**	**N14**	**N15**	**N25**	**N23**	
5	**……**	**……**	**……**	**……**	**……**	**……**	**……**	**……**
6	**……**	**……**	**……**	**……**	**……**	**……**	**……**	**……**

Figure 28: Failure Table

The occurrence of each node in the FT is sorted, and an occurrence table is made for the following step (Figure 29).

Rank	*ID*	*Occurrence*
1	**N1**	**17**
2	**N0**	**14**
3	**N3**	**13**
4	**N6**	**6**
5	**N9**	**3**

Figure 29: Occurrence Table

The second step is to focus on the failure patterns, which are chins that have failed multiple times and are listed in the FT. For example, the failed chain is:

N0 -> N1 -> N2.

This chain failed five times and is listed in the FT with an occurrence of 5-the highest ranked failed chain in the FT.

Rank	*Pattern*	*Occurrence*
1	**N0->N1->N2**	**5**
2	**N9->N10->N12**	**3**
3	**N8->N12**	**2**
…	**…**	**..**

Figure 30: Occurrence Table for Patterns

The third step is divided into two sub-steps. The first sub-step is to match the occurrence and the patterns. For example, the pattern, N0->N1->N2, has the most number of occurrences, and N1 has the top rank in the occurrence table (Figure 30). The rank of N1 is higher than that of N0 and that of N2.

The second sub-step is to eliminate the number of occurrences in the occurrence table. Node N1 is the most probable abnormal node. Therefore, the number of occurrences of N0 is subtracted from the number of occurrences of patterns, 5, in the updated occurrence table (Figure 31).

Rank	*ID*	*Occurrence*
1	**N1**	**17**
2	**N3**	**13**
3	**N0**	**9 (14-5)**
4	**N2**	**1 (6-5)**
…	**…**	**…**

Figure 31: Updated Occurrence Table

Chapter 3. Simulation

This chapter will discuss why the RCM is an excellent self-adjusting model on the basis of results obtained using a simulation tool.

3.1 Considerations

A simulation tool was used to demonstrate the relationships between the number of neighbors for each node, the number of total nodes in the P2P community, and TTL in finding the resource. The following questions must be answered: When the maximum number of neighbors for each node is set at what number should the TTL be set to get a reasonable searching result? Moreover, when the TTL for each node is set, at what number should the maximum number of neighbors be set to get a reasonable search result?

It is important to demonstrate RCM's effectiveness and efficiency (not considering the malicious nodes). When the total number of nodes is set, the maximum number of neighbors for each node and the TTL are important. It is essential to make a reasonable group of nodes by using this variable on the basis of the first simulation statistics. It is necessary to keep in mind that the algorithm can go through almost every node in this P2P community. The useful data is the percentage of the nodes visited out of all the nodes when a certain request is sent out.

In addition, if the percentage nodes that have the resource is sufficient, many potential routes may be found to the destination node in order to download the resource.

It is important to set up the connection between these possible routes and the maximum number of neighbors and the TTL along all possible routes.

In addition, it is necessary to demonstrate that the RCM is more powerful than models (considering the malicious nodes in the network). The main difference between the RCM and others models is that in the RCM, the credibility of the neighbors is updated after only one transaction. Hence, it is necessary to update the neighbor list and the RCM in order to use the RCM as a model for a pure P2P network. The main data have the successful percentage of the possible routes to the final destination considering the malicious nodes effect. Others' algorithms that did not consider the updating will have the same success percentage as time goes on. For this reason, the RCM has numerous advantages for increasing successful download rates when nodes become more cautious about the malicious nodes in the network.

The P2P network is dynamic because some nodes may leave the network and new nodes may join. For this reason, the RCM can make each node a global (or sub-global) view of the whole network's malicious group. By doing so, peers can decide whether a node is good or malicious by asking or by recalling past transactions with other peers. Neighbors are less important in this judgment process. The data compared here will be further analyzed in the following section.

3.2 The First Simulation and Data Analysis

The most valuable information that is necessary to collect is the change in the malicious nodes' impact on the Resource Chain Model. The percentage of the number of successful chains out of all resource chains produced in one transaction when using the RCM. Three groups of data were used to generate the result.

First, the simulation tool was run without the reputation control mechanism. The total number of nodes was 500, and the number of malicious nodes was randomly selected as 50 in the P2P community. The total number of transactions was 100. The numbers of all the successful resource chains were recorded. Figures 32, 33, and 34 show the percentages through the transactions, and the graphs at the bottom in Figures 32, 33, and 34 show the results obtained without using the reputation control system. The straight black lines inside the graphs at the bottom show the trend in the rate of successful downloads.

The simulation tool was then run using the P2P reputation control method. The data are plotted as the graphs at the top in Figures 32, 33, and 34, and the straight black lines inside the graphs at the top show the trend in the rate of the successful downloads. The vertical lines represent the percentage of successful downloads and the horizontal lines represent the number of transactions.

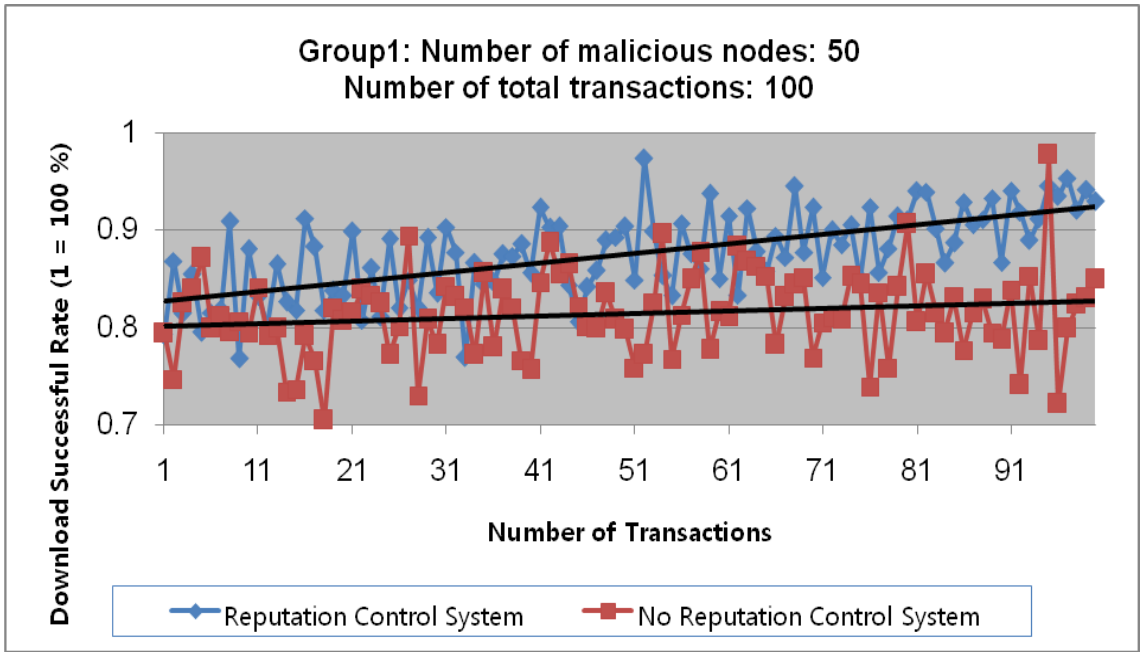


Figure 32: Group 1 Data Analysis Result

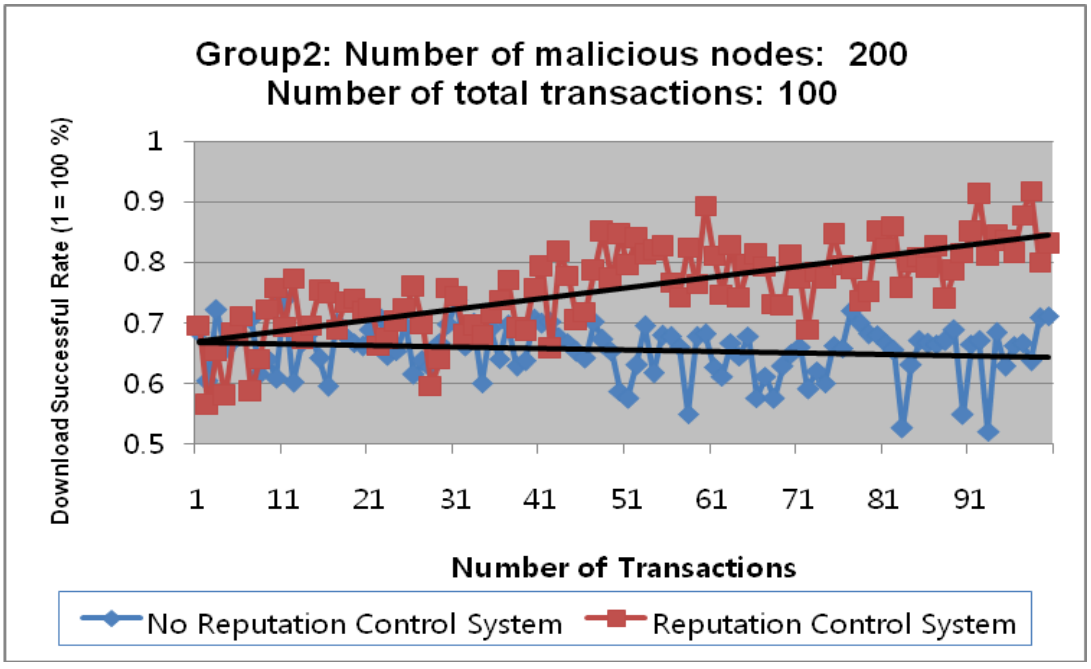


Figure 33: Group 2 Data Analysis Result

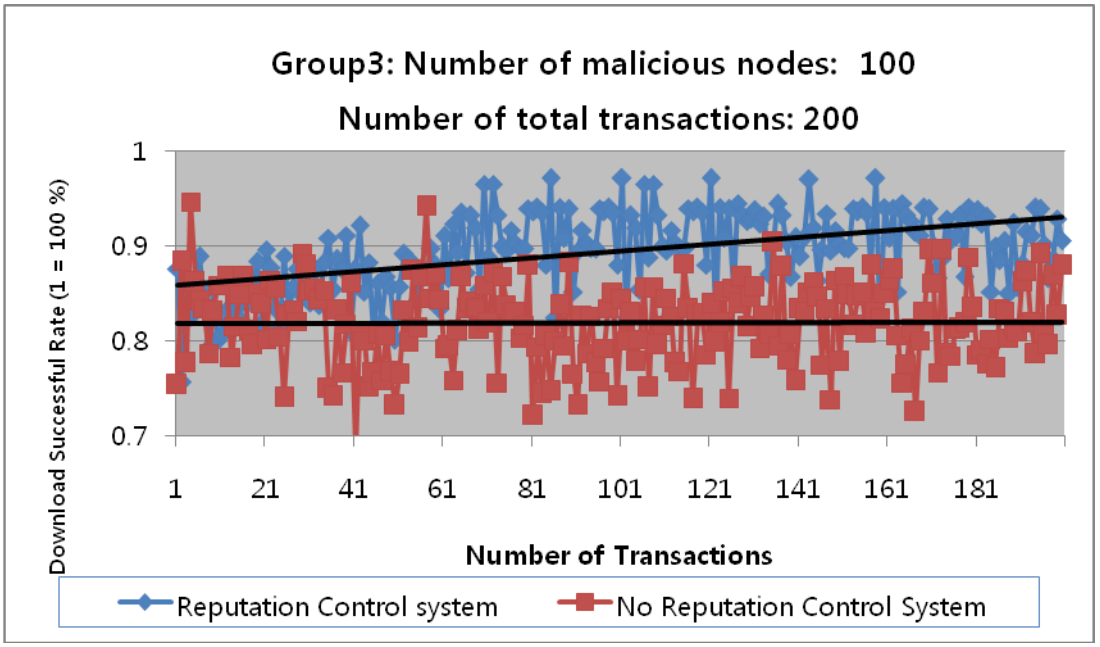


Figure 34: Group 3 Data Analysis Result

The results clearly show that the self-adjusting reputation control helps the system to automatically decrease the impact of malicious users. Comparing the data for groups 1 and 2, as the number of malicious nodes increases in the system, the greater the effect the RCM has in increasing the trust rate of the whole system.

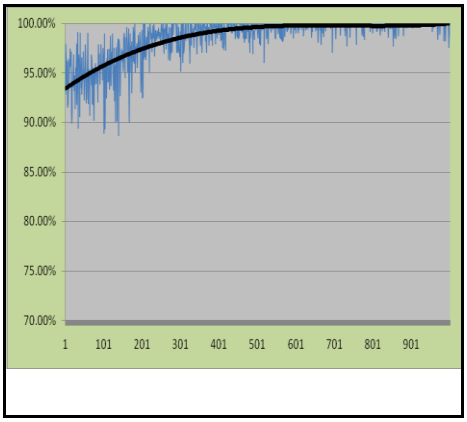
More transactions are called and more malicious nodes take part in the transactions in group 3 than in group 1. This suggests that all nodes in the system will be able to identify the malicious nodes. The RCM can use this kind of data updating to generate a better communicaion evironment for all the nodes in this system.

3.3 The Second Simulation and Data Analysis

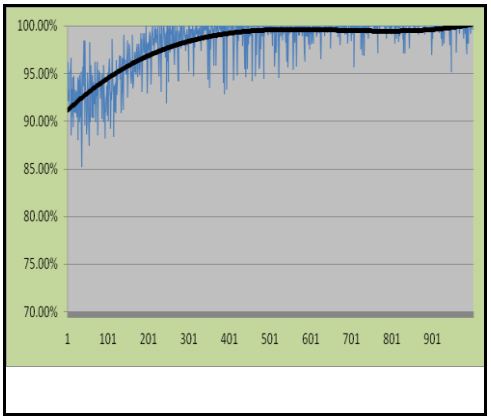
Figure 35 shows the simulation tool used for collecting five different groups of data. Although the total number of nodes was the same for each group, the number of

malicious nodes was different. For each test, the system ran 1000 transactions, and for each transaction, it was necessary to collect data about all the possible downloading paths and to calculate how many paths were effective resource-chains; that is, those chains that are not affected by the malicious nodes in the system. It is important to record the number of effective paths as a percentage of the total number of paths because it is an indication of the level of security of the current system.

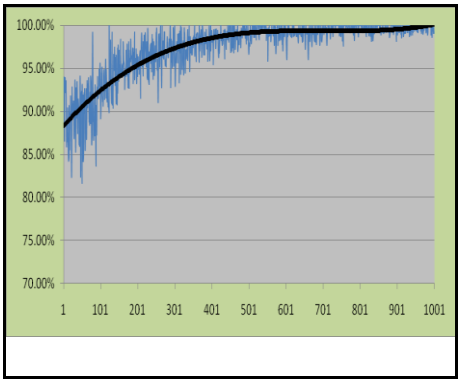
As more transactions happen, more malicious actions and even some malicious nodes are identified. The malicious nodes are now less of a threat to system security because they have been identified and have even been blocked by other nodes when the system used the dynamic reputation control mechanism. Figure 35 shows the test results.



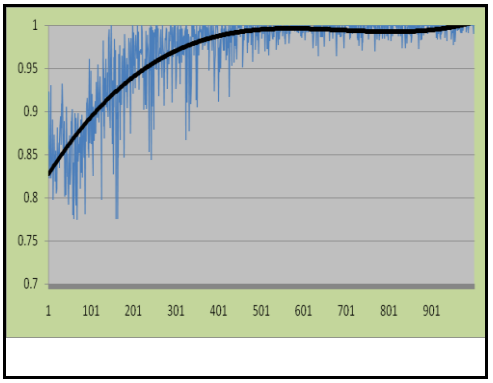
a (Number of Malicious Nodes: 30)



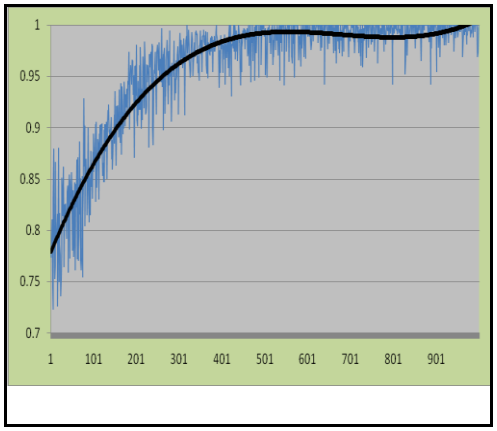
b(Number of Malicious Nodes: 40)



c(Number of Malicious Nodes: 50)



d(Number of Malicious Nodes: 75)



e(Number of Malicious Nodes: 100)

Group	Total Node	Malicious Nodes	Transactions	Figure
1	500	30	1000	(a)
2	500	40	1000	(b)
3	500	50	1000	(c)
4	500	75	1000	(d)
5	500	100	1000	(e)

Figure 35: Rate of Successful Transactions

From the five groups of data collected and the analysis of the trend curves (Figure 35), the RCM is clearly doing a reliable job supporting the P2P community in increasing network security. The RCM has the following there advantages.

It increases the security of the P2P network without central control. The best-fit curves of the data, which represent the trend in the security level of the system, can be approximated by third-order polynomial functions.\

In all the diagrams, it is clear that the number of successful downloads always increase with the number of transactions. The number of successful downloads converges to 100% of the successful transactions in the P2P community, which represents maximum security of the system. This means that the RCM really supports the P2P network by automatically increasing the system security without any central control.

The RCM also rapidly self-adjusts. In all the trend diagrams, it is clear that after the number of transactions reaches about 500, the success rates are all close to the maximum success rates. This suggests that the security rate of the system converges to the maximum rate after only about 500 transitions for a 500-node-scale P2P community. After each node sends out a request and finishes one transaction, the system almost adjusts itself to achieve a much better level of system security. It is necessary to keep in mind that such a fast convergence of RCM's self-adjusting ability is a reliable metric of the usefulness of the RCM.

The RCM also increases the stability of the system. The best feature of the data analysis is the number of malicious nodes that exist in the system. Figure 35 shows that after about 500 transactions, the success rates converge to the maximum success rate. Generally, as the number of the malicious nodes in a P2P network increases the overall

system security decreases. However, by selecting data groups carefully, it is difficult to change the RCM's self-adjusting ability and security. The RCM updates the chain credibility instead of updating the normal node-to-node credibility. There are several great advantages of using the RCM to increase the system stability.

3.4 Comparison with Complaint-Only Model

As is well known, the complaint-only model concentrates on the credibility relationship between nodes instead of on the chain view of all the resource chains. In this model, the total number of nodes is 500, the same as the number of P2P community nodes. Five hundred transactions were run to ensure that the data in both experiments were gathered under similar conditions. After about 500 transactions, the RCM's performance had converged to almost the level where maximum security is achieved. After all the data were plotted, a best-fit curve was drawn to represent the overall trend of the security rate, which gradually increases. Figure 36 shows a comparison of the two models.

From the comparison diagram, the difference in the performance of both models is obvious after 500 transactions. For the RCM, the system security is quite good, and the trend curve shows the improved system security after 500 transactions. However, the Complaint-Only model clearly requires more than 500 transactions before any appreciable benefits to system security are observed.

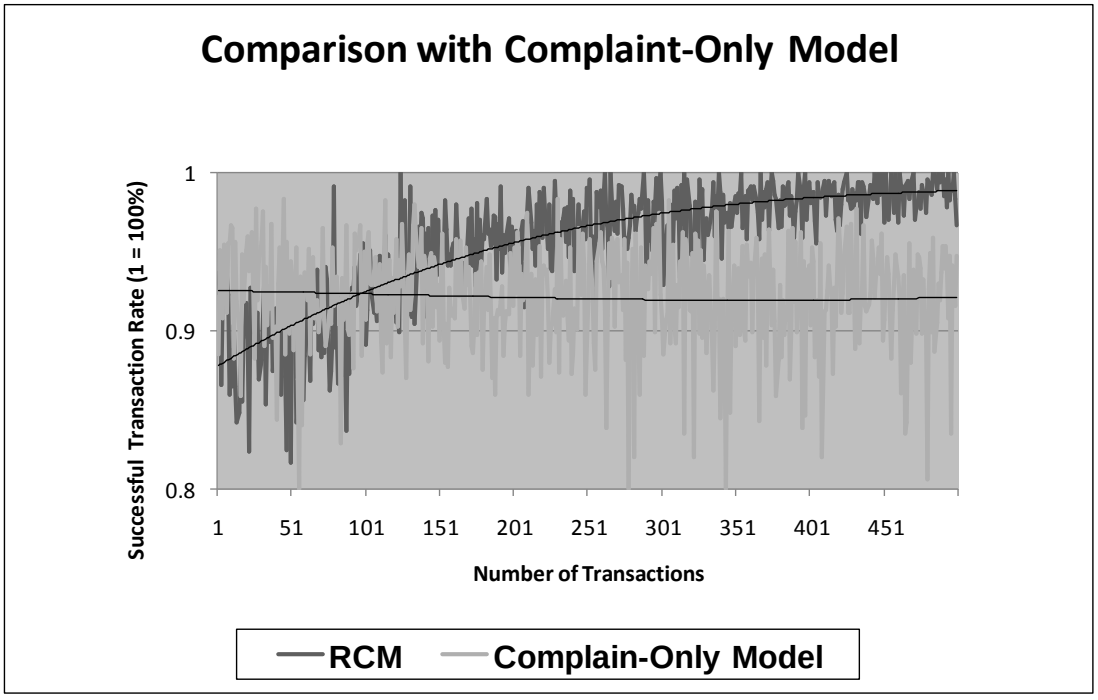


Figure 36: Comparison with Complaint-Only Model

3.5 Comparison of RCM Results Obtained Using Different Numbers of Nodes

It is necessary to compare the RCM results obtained using different number of Nodes in order to improve the RCM. Figures 37, 38, 39, 40, and 41 clearly show that the different number of nodes can affect RCM more efficiently.

3.5.1 Comparison of RCM Results Obtained By Increasing Total Number of Nodes

The following simulation was created by increasing the total number of nodes in the RCM.

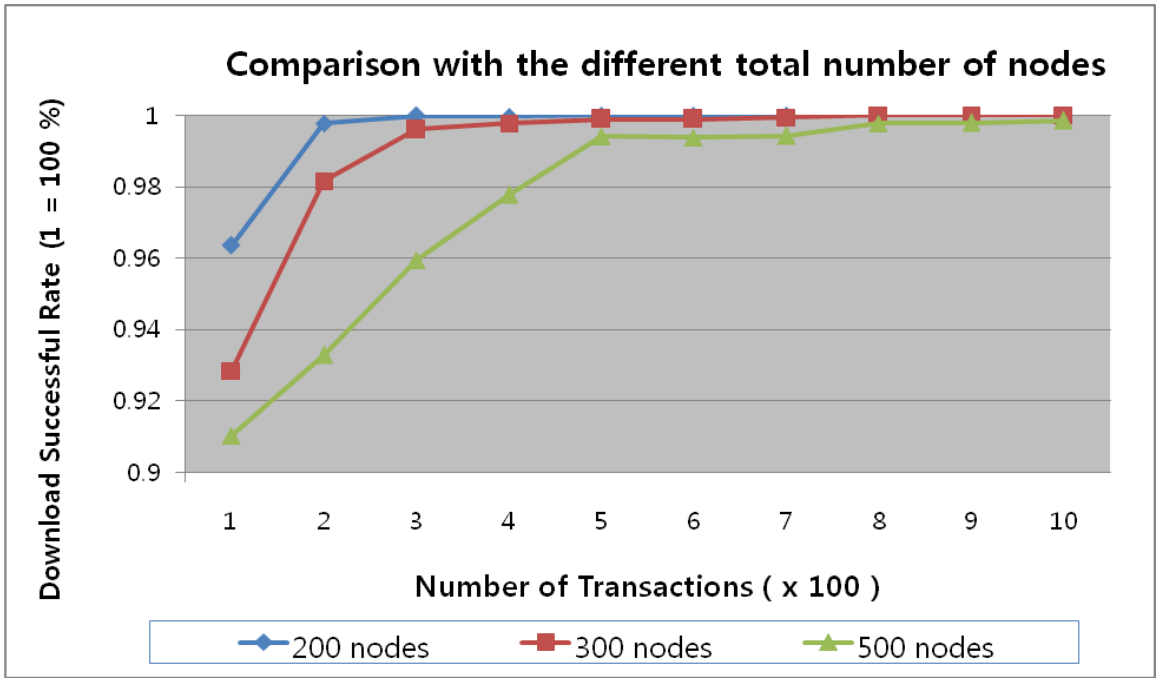


Figure 37: Comparison of RCM Results Obtained By Increasing Total Number of Nodes

The Y-axis represents the rate of successful downloads, and the X-axis represents the average number of transactions. The value of "1" on the X-axis represents a 100% successful download. When the total number of nodes is increased, each graph converges to a 100 % successful download.

From the point of view of efficiency, this is an extraordinary graph because real-world real P2P networks have numerous nodes. In all cases, when the total number of transactions is 800, the rate of successful download reached the maximum value as shown in Figure 37. The percentage of malicious nodes in this simulation was 10%.

3.5.2 Comparison of RCM Results Obtained By Increasing Number of Neighbors

This simulation was created by increasing the number of neighbors for each node in the RCM.

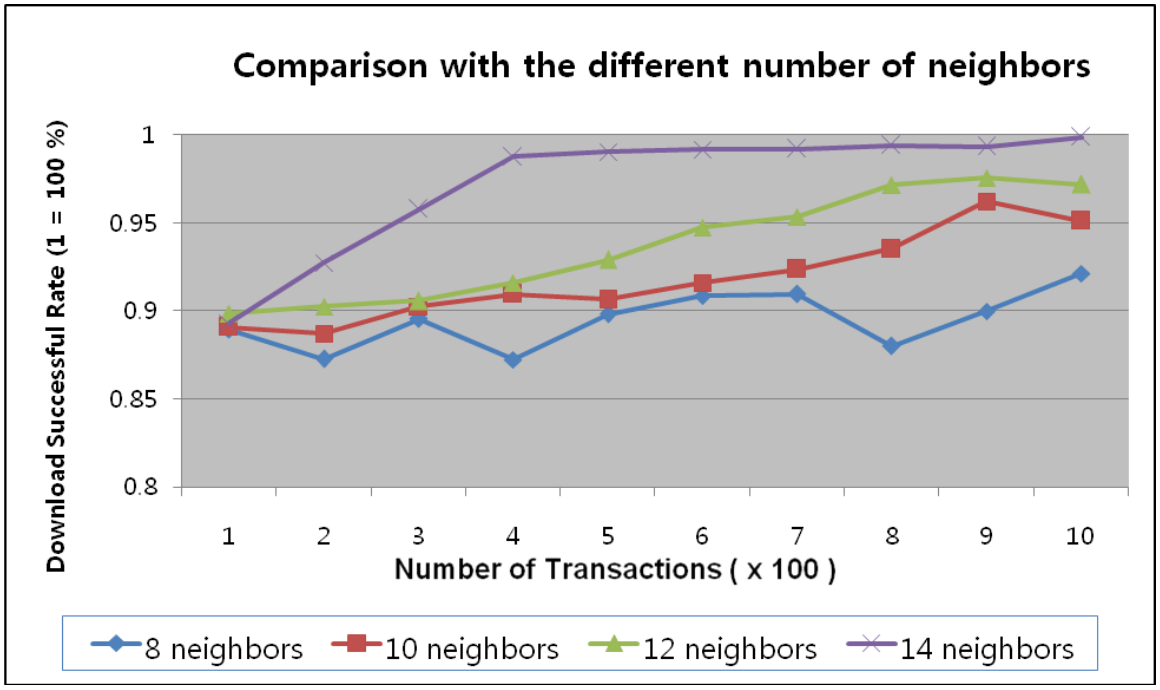


Figure 38: Comparison of RCM Results Obtained By Increasing Number of Neighbors

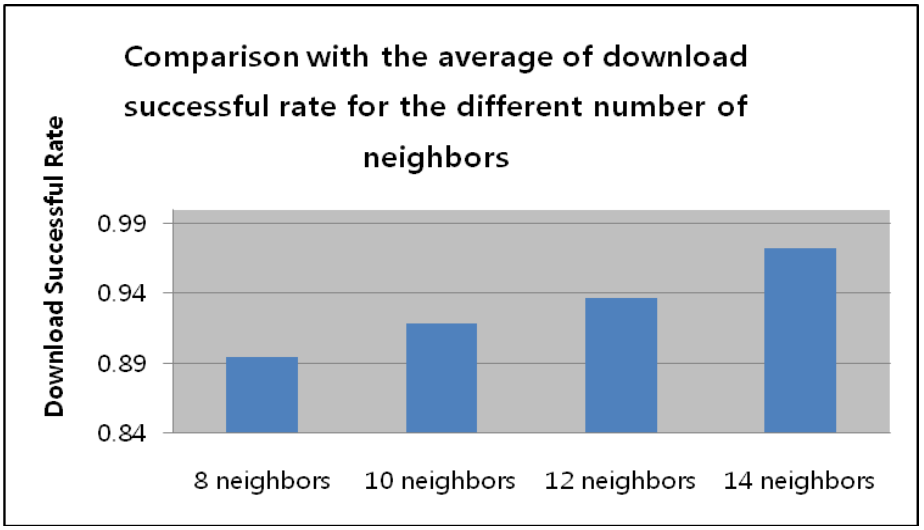


Figure 39: Comparison of Average RCM Results Obtained For Different Numbers of Neighbors

The Y-axes represent the rate of successful download and the X-axes represent the number of transactions involved. The value of "1" on the X-axis represents the average rate of successful downloads of the first 100 transactions. When the number of neighbors for each node in the RCM is 14, the self-adjusting ability and the rate of successful download are maximized as illustrated in Figures 38 and 39. These results show that the effectiveness and efficiency of the RCM increases with an increasing number of neighbors.

3.5.3 Comparison of RCM Results Obtained For Different Total Number of Transactions

Data were gathered by increasing the total number of transactions. The results showed that the self-adjusting ability of the RCM increases with an increasing number of transactions (Figure 40).

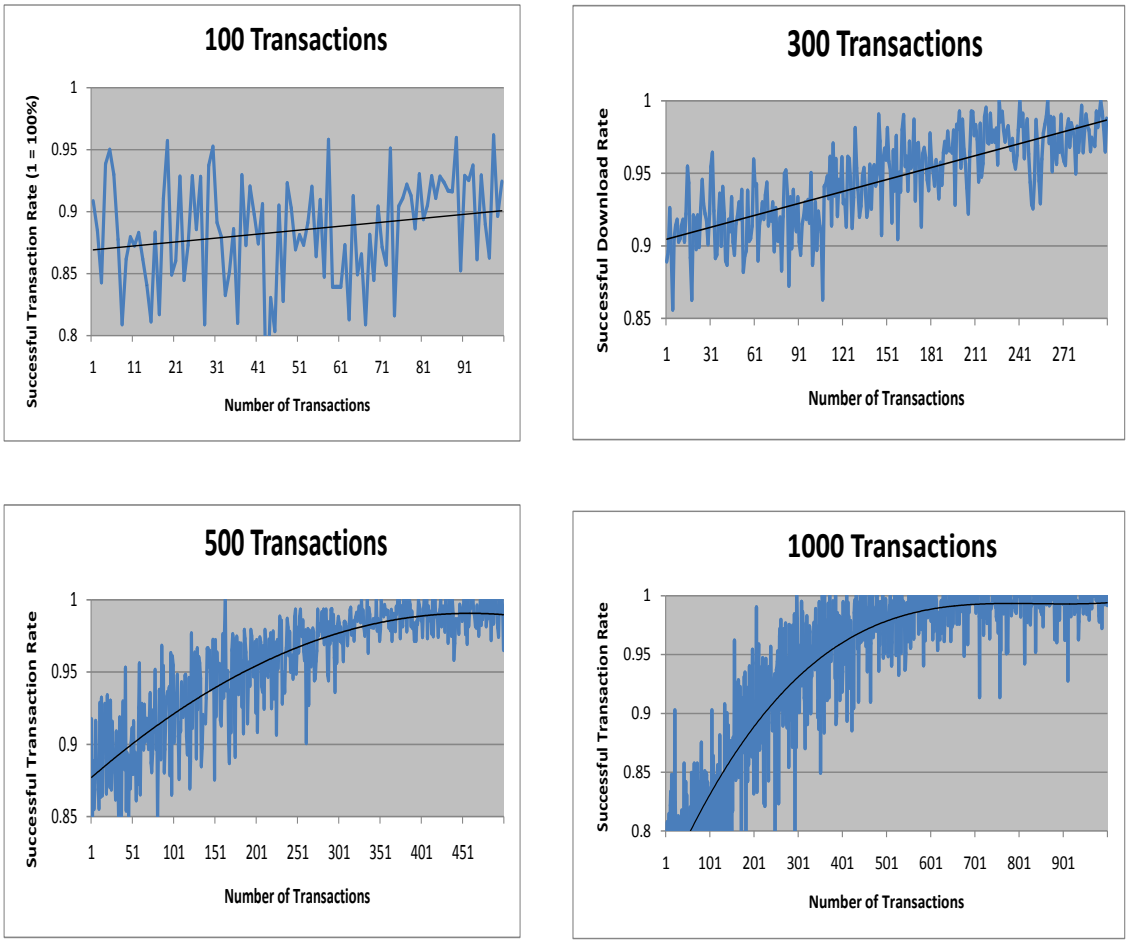


Figure 40: Comparison of RCM Results Obtained For Different Total Number of Transactions

3.5.4 Comparison of RCM Results Obtained For Different Number of Resources

This simulation was created by increasing the number of resources (files) for each node in the RCM. Figure 41 shows that when the number of resources in the RCM is 20, the self-adjusting ability of the RCM is maximized. This result shows that the effectiveness and the efficiency of the RCM increased with an increasing number of resources.

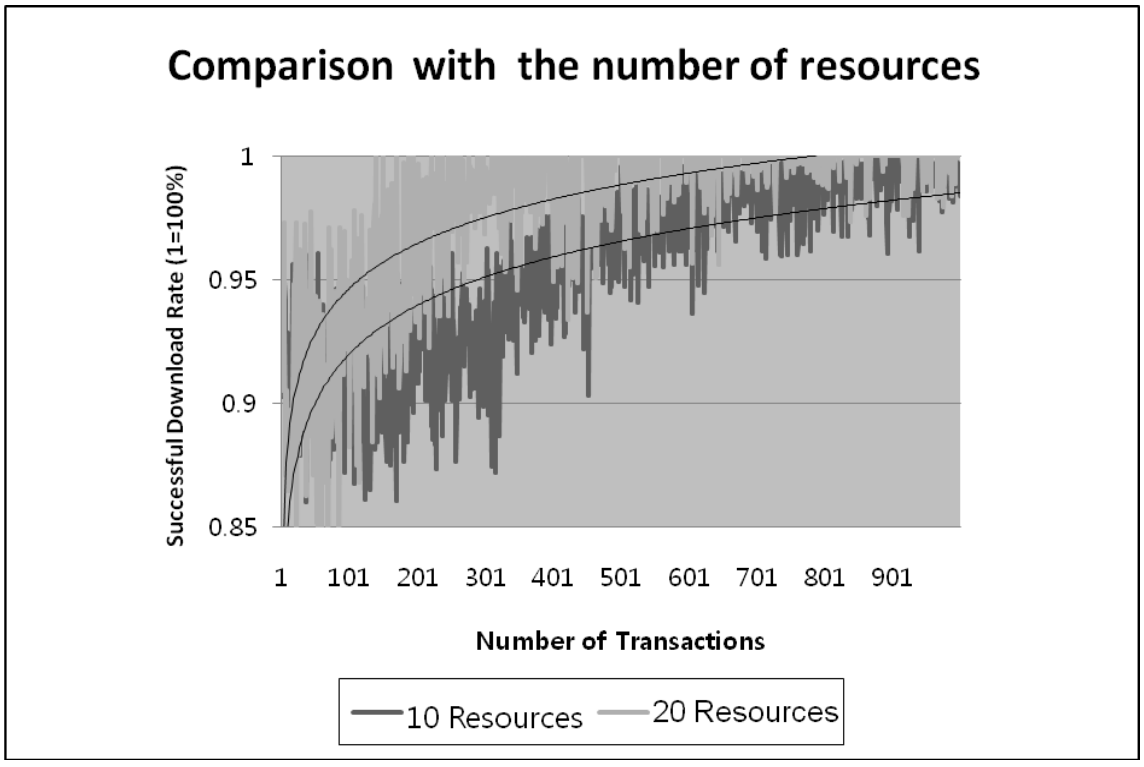


Figure 41: Comparison of RCM Results Obtained For Different Number of Resources

3.6 Simulator

Before starting to make the simulation tool, the researcher posed several questions such as “When can researchers set the number of maximum neighbors for each node?” and “What is the maximum number required for the TTL to get a reasonable search result?” It was also necessary to demonstrate the effectiveness and efficiency of the RCM algorithm.

3.7 Evaluation

From the test results, it is clear that the dynamic reputation control system–RCM helps the system to achieve better security after a number of transactions. From each trendline, which describes the trends in the rate of system security, the community’s security gradually increased. As a result of the improvement of the transformation, the trendline is much faster. The RCM has great advatages over other models, and the RCM greately increases the system security by automatically self-adjusting the credibilities of nodes in the system on the basis of chain concepts.

3.8 Simuation and Data Analysis for Identifying Malicious Nodes

This simulation was performed to prove that the RCM improved the identification of malicious behaviors.

3.8.1 The Steps of the Simulation for Identifying Malicious Nodes

The first step is node generation. The simulator needs the identification of each node and the status for each node must be identified as malicious or good. Additionally,

the nodes in the system have initial resources. Most importantly, the simulator needs a starting neighbor list in which each neighbor starts with a 0.5 credibility. Nodes are recorded in a log file after they have been identified.

The second step of the simulation is to collect the failed transaction data with the starting node, the destination node, and all the nodes in between. The next step is to analyze the data in the failure transaction table. The main function of the failure transaction table is to forward failed transaction data. No doubt each node has a failure transaction table. The table records information about the whole route from the starting node to the destination node.

The analysis approach include the following steps. For exactness, the starting node and the destination node must receive computational consideration. To take a hypothetical example, each node gets a 100% affecting factor, which means that the researcher takes full computational consideration. By contrast, the nodes in between the starting node and the destination node receive less computatinal consideration. As an illustration, the starting node and the destination node receive twice the computational consideration as the in-between nodes. The last step in the analysis approach is to calculate the sum of the factors occuring for a certain node in the failure table. If the credibility is larger than the threshold, the node is malicious. The last step of this simulation is to identify the malicious nodes. If nodes have an occurrence factor larger than the threshold, the nodes must be malicious. Also, the simulator maintains a list of malicious nodes.

3.8.2 Simulation and Data Analysis of the Enhanced Resource Chain Model

In the enhanced resource chain model, it is essential to add the trust factor (TF) for each node in the enhanced resource chain model. Each node has its own TF, which is used to classify the nodes as normal, malicious, and camouflaged. There are two advantages in using this type of simulation.

The first advantage is a high percentage of identified “malicious nodes” and “malicious collective.” All “malicious nodes” and “malicious collectives” always lie during transactions. It is necessary to identify “malicious nodes” and “malicious collectives” because such nodes have a greater chance of attacking the P2P network than have normal nodes or “camouflaged nodes.” Therefore, if a high percentage of malicious nodes is identified, the P2P network is more reliable.

Figure 42 shows that the recognition percentage increases quicker at the middle of the curve than the slope of the secant, which increases slower than the middle of the curve after about 6000 transactions. However, the trend line of the graph increases constantly during the time of the transactions. It is enough to prove that the percentage of recognized “malicious nodes” and “malicious collectives” is sufficient to maintain the P2P network security. The vertical axes represent the percentage of recognition rates and the horizontal axes represent the number of transactions involved (Figure 42). In the simulation, the simulator has the defined numbers (Node_Total =10000, Time_to_Live=5, Transaction_Total=10000, Max_Neighbor_Num=8).

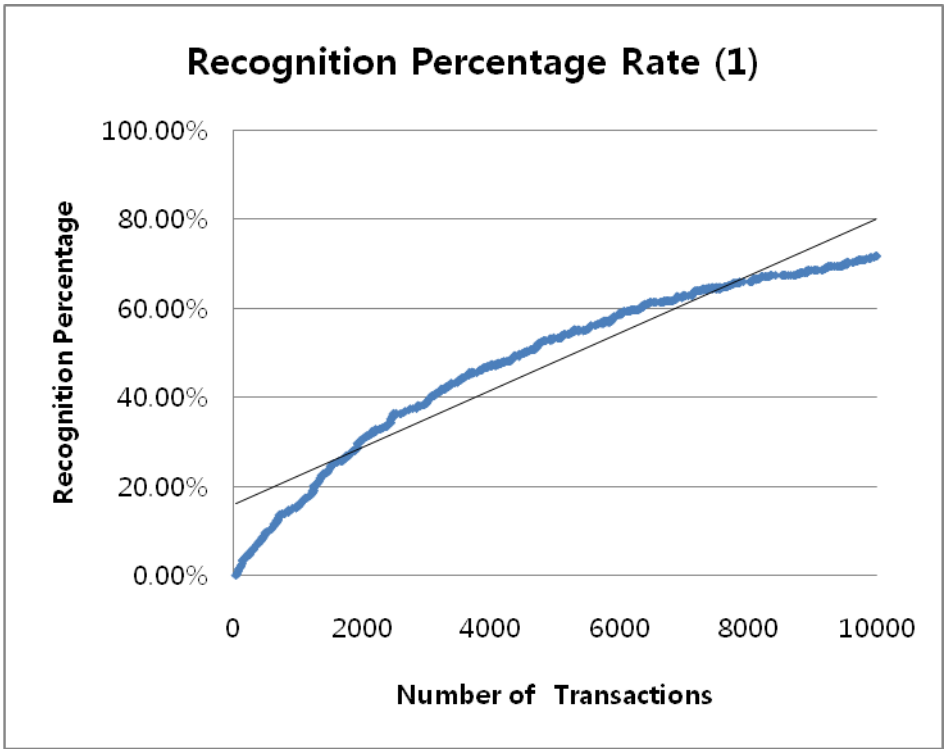


Figure 42: Recognition Percentage Rate (1)

The second advantage is a high percentage of identified nodes of all kinds of abnormal nodes with "camouflaged nodes." While "malicious nodes" always lie, "camouflaged nodes" sometimes tell the truth. In Figure 43, it is clear that the enhanced resource chain is powerful enough to increase the P2P network security. In Figure 43, the simulator results have the same scale as those shown in Figure 42.

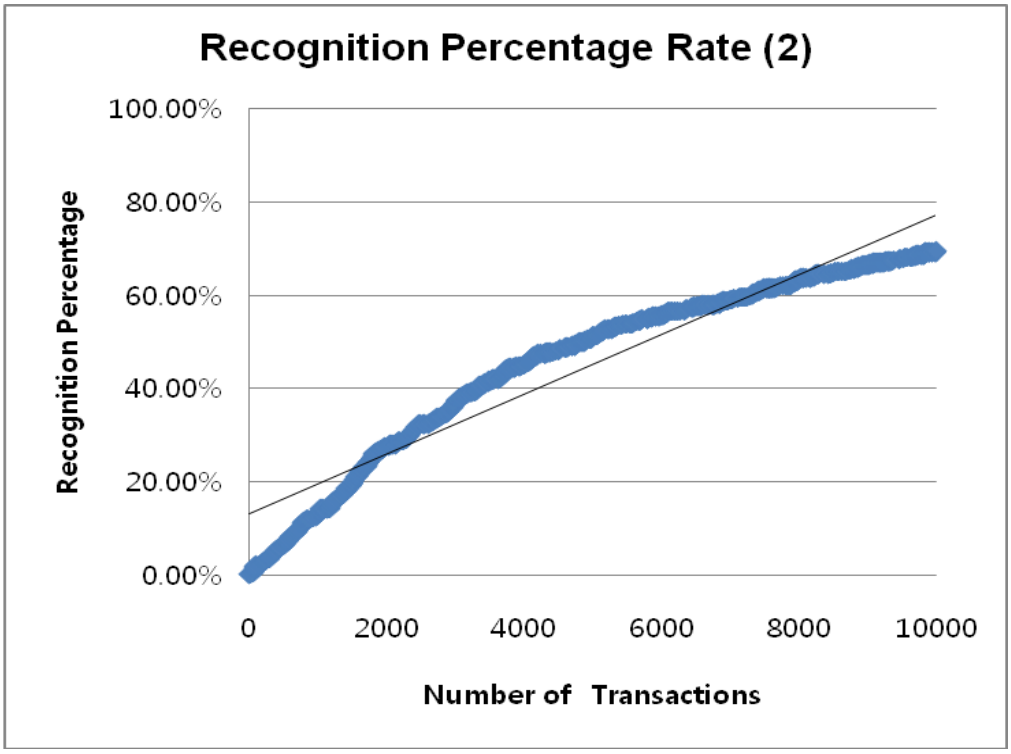


Figure 43: Recognition Percentage Rate (2)

The next simulation is to compare the rate of download success of the system using the enhanced resource chain model and that of the system without using the enhanced resource chain model. By using the same scale of the simulation results shown in Figure 42 and 43, it is possible to record the success rate of 10000 transactions. The diagram on Figure 44 shows the change in the rate of download success of the system without using the enhanced resource chain model. The diagram of Figure 45 shows the change in the rate of download success of the system using the enhanced resource chain model. The results show that the rate of download success of the system using the enhanced resource chain model is much higher than that of the system without using the enhanced resource chain model.

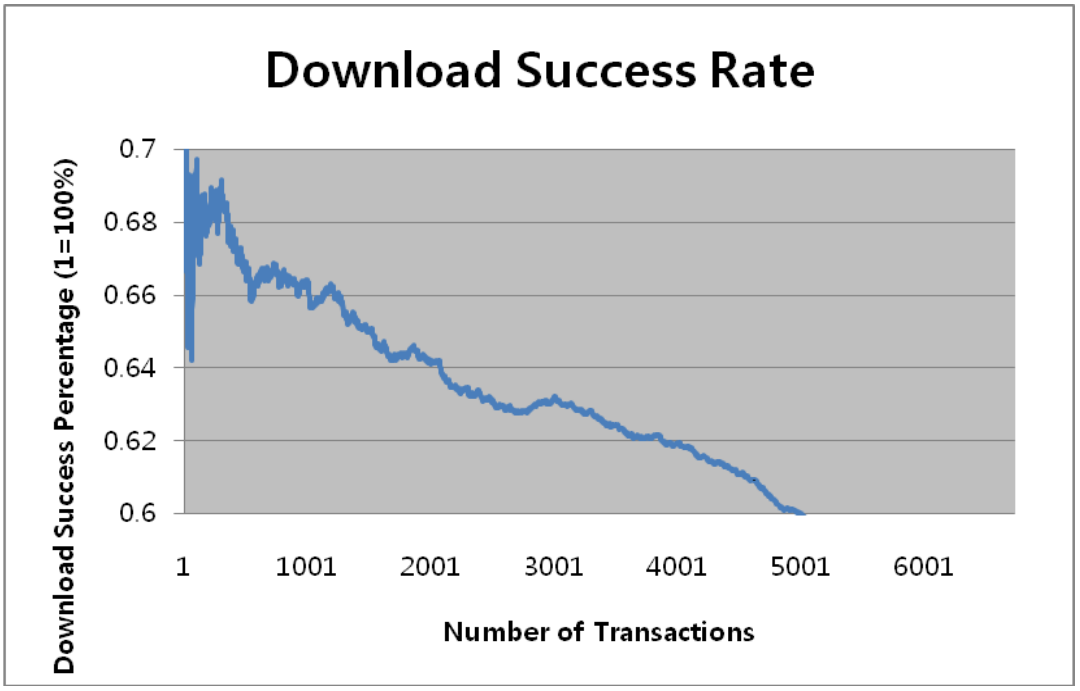


Figure 44: Rate of Download Success without ERCM

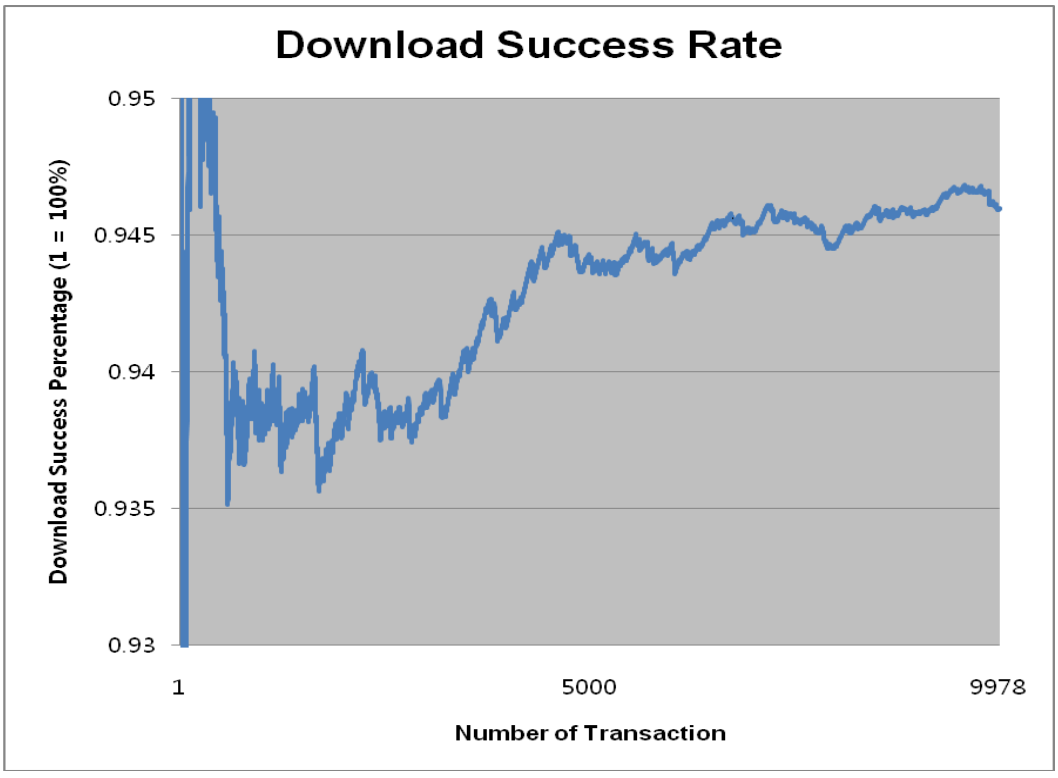


Figure 45: Rate of Download Success with ERCM

The vertical axes represent the percentage of successful download rates, and the horizontal axes represent the number of transactions involved (Figure 44, 45).

Chapter 4. Conclusion

It should be concluded, from what has been said above, that by using RCM, the P2P reputation problem can be solved and distributed efficiently. The RCM can find the malicious resource chain after each transaction and weaken the whole chain to prevent it from happening again. In that way, the RCM helps everyone in the P2P network is honest and benefits the whole P2P community. Such a basic idea of the RCM can well serve the P2P file sharing system.

The result derives from the following facts: the RCM can find the resource chain with malicious nodes after each transaction and weaken the whole chain to prevent it from happening again. It is essential to promote each other's transaction reputation if the transaction is successful. Therefore, the RCM can be applied to a P2P file sharing system to enhance the security and automation adjustment of that system. Because of the automatic information collection and reputation updating according to different system environments, the RCM can be well applied to such communities as a technical method to make the P2P world more trustful.

The most advantages of this model:

1. The searching process of the best resource candidate is a heavy task. In the RCM, this task is assigned to Node N, his neighbors, and his neighbors' neighbors and so on. Such a load balance can greatly provide the network scalability and process speed.
2. The RCM concentrates on the chain effects after one transaction happens; to strengthen the resource chain of a successful transaction is the point while weakening

an unreliable resource chain.

3. It should also be added the RCM maintains a good enough neighbors' list and dynamically adjusts their credibility according to transaction results. In the real society, other opinions can be considered according to their deeds dynamically.

The feasibility of this model:

The RCM can be applied to a P2P file sharing system to enhance the security and automation adjustment of that system. Because of the automatic information collection and reputation updating according to different system environments, The RCM can be well applied to such communities as a technical method to make the P2P world more trustful.

Some Future Works

There is still future research to do with the RCM. One thing to note, if peers can directly know the resource location, peers do not ask for more requests. On the contrary, this may result in malicious activities. Another problem is different resources, with different credibility problems. Though it is good to assign each node a credibility factor, it is also true that for the same node, different resources have different credibility. That is true when a math professor supplies many useful mathematic materials while the materials of chemistry or biology provided by may not as credible. Therefore, in the future studies, these credibility problems must be examined.

Summary

This dissertation approaches the new model of P2P reputation-based systems. As is well known, computer security has recently been given more attention; many mechanisms have been developed to increase P2P security like encryption, sandboxing, reputation, and firewall. Among those technologies, reputation mechanism as an active method is especially useful to automatically record, analyze and adjust peers' reputation, trust, and histories among the different peers. This method is suitable for the anonymous and dynamic P2P environment. Therefore, reputation-based trust mechanism is a secure reputation system.

Many researchers have given a lot of effort to this reputation-based system area, and some of them have made good theoretical models. One problem is that such models concentrate only on the relationship between the node and its direct neighbors. Usually when peers want to find a resource, peers will get the resource path through many peers and peers' neighbors. Such a resource path carries the most information of the future resource searching.

In this dissertation, the new reputation-based trust management model is presented to prevent the spread of malicious content in the open community. The main idea of the RCM is using a routing table to record the information of nodes' credibility and their recommending nodes' credibility of feedback.

From the test results, compared with the system without the reputation control, the self-adjusting system can help the system itself in automatically decreasing the impact of the malicious users in this group. Such a basic idea of the RCM can well serve the P2P file sharing system. Therefore, using this model can help us efficiently find the best and

safest resource location and decrease the number of malicious transactions.

APPENDICES

Appendix A: The Simulation Tool Guide

The simulation tool was written using C# programming language. The simulation tool needs a number of nodes for configuration. One node has between 1 to 5 resources at first. When the total resource number is set, the Time-to-live is defined to 5. Here is Figure 46, which shows a screenshot of the simulation tool.

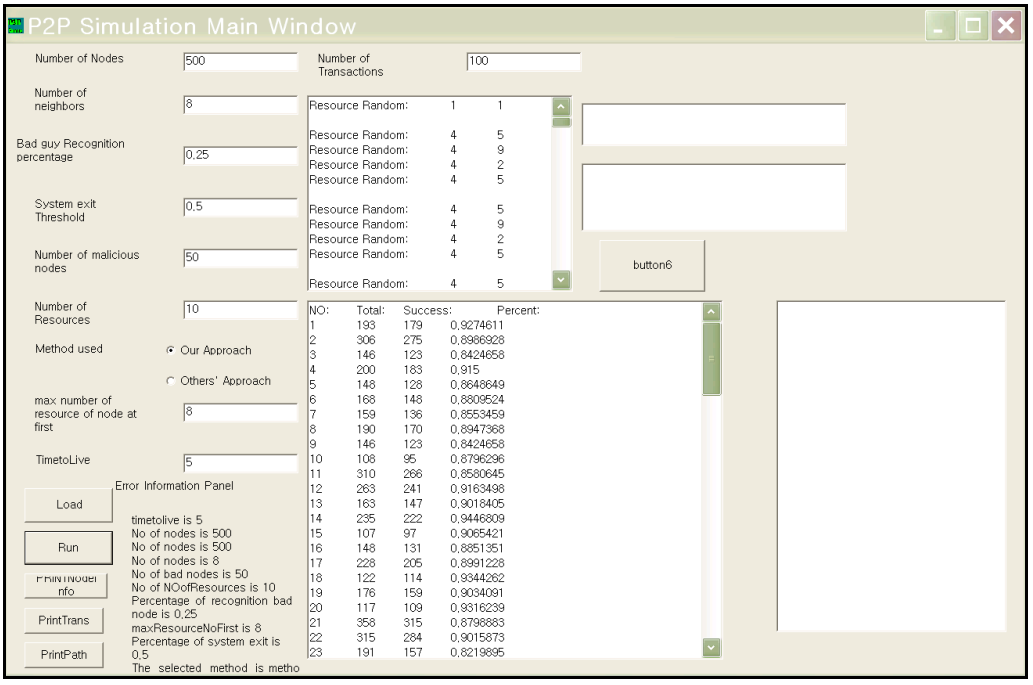


Figure 46: A Screenshot of the Simulation Tool

Default numbers of total nodes, neighbors, malicious nodes, and resources for each node appears in Figure 46. There are many items to be changed in the simulation tool. The first item is the number of total nodes. The default number of total nodes is 500. The next item is the number of neighbors for each node. Also, it has a default number of neighbors, which is 8.

The following item is the malicious node recognition rate. This means if one node out of four nodes reports a node is malicious, the node must be malicious in the RCM. “System exit threshold” indicates that if crediblity of a node is lower than the threshold value, the node must be out of the system. The number of malicious nodes can be changed in the tool.

The number of resources means each node can have a different number of resources. Then a user can select one of the options, which are our approach and the approaches of others. In other words, our approach has a coneection with the RCM and the approaches of others refer to the complaint-only model. In order to find the reources, it is necessary to decide the maximum number of resources at first in the RCM. The TTL is the next item.

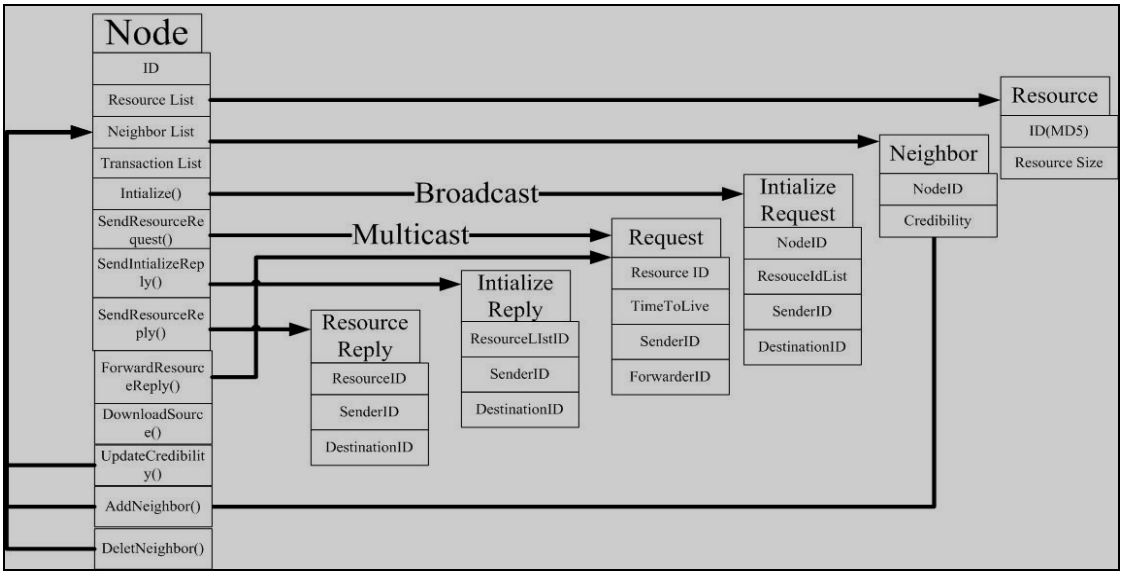


Figure 47: The Architecture of the simulator

As Figure 46 and 47 indicate, there are many functions in the simulation tool. The important point in the architecture is the updateCredibility(). This function is the main part of the simulator to update credibility for each node.

Name of Method	Function
generateNodes()	To generate all the nodes we need
initializeNeighbors()	To randomly generate all neighbors for each node
updateNeighbors()	To update the neighbor list which can delete malicious nodes and add new nodes
generateResources()	To generate resources randomly
SendResourceRequest()	To simulate the resource request process

Figure 48: Functions of Each Method

These are the previous P2P simulators. GPS means a general purpose simulator for a P2P network [37]. There are three simulation core components [25]. The first is the simulation engine, which has a discrete event simulation engine with global queue and an event scheduler. The simulation engine uses a heap structure with insertion and deletion times which are represented by O(log(n)). The second component of GPS is a agent. Each node contains more than one agent (lower layer and higher layer). The last component is a document, which has basic information about the file size. Another kind of P2P simulator is the P2PSIM [38], which has a free, multi-threaded, discrete event simulator in order to evaluate, investigate, and explore P2P protocols. It runs in several UNIX-like operating systems. Moreover, P2PSIM is part of the IRIS project, MIT [26].

Appendix B: The Simulation Tool Guide for the Second Phase

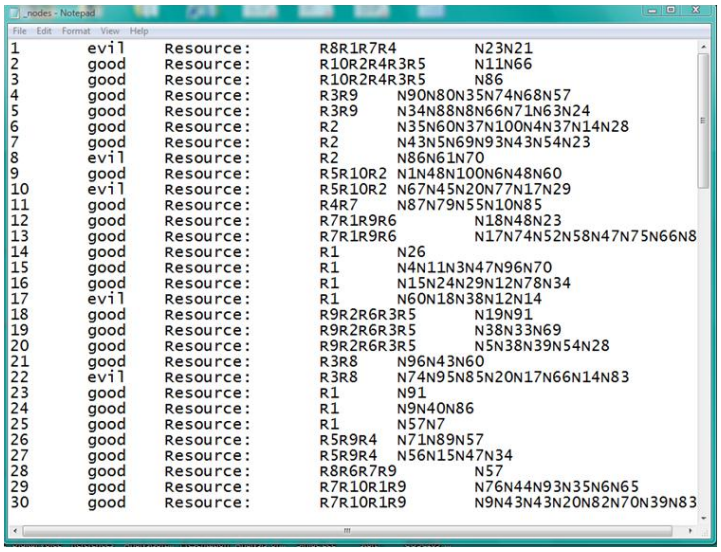


Figure 49: A Screenshot of the simulation for the second phase

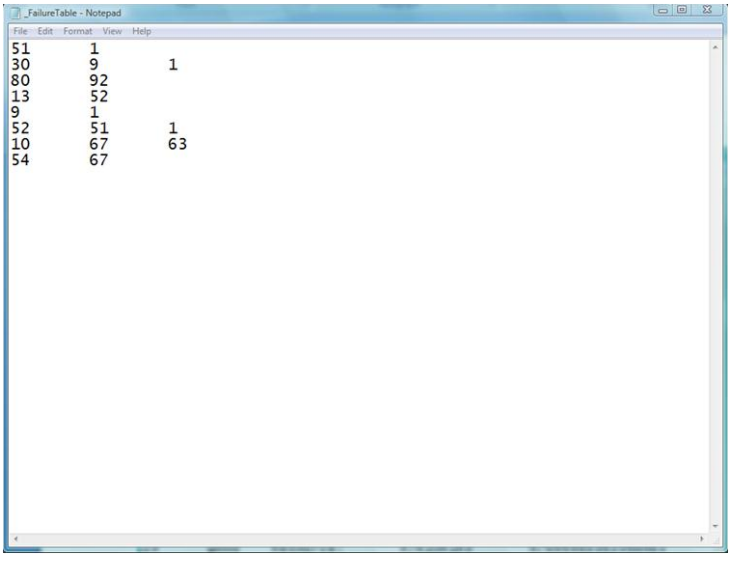


Figure 50: A Screenshot for Failure Table

Name of Method	Function
generateNodes()	To generate all the nodes we need
initializeNeighbors()	To randomly generate all neighbors for each node
generateBadNodes()	To randomly generate all malicious nodes
Request()	To simulate the request sent by one node
checkResource()	To judge if a resource is in the node's list

Figure 51: Functions of Each Method

References

[1] S. Lee, S. Zhou and Y. Kim, "P2P Trust Model: The Resource Chain Model," SNPD 2007, Eighth ACIS International Conference on Volume 2, Issue, July 30 2007-Aug. 1 2007 pp. 357 – 362. http://tiger.towson.edu/~slee5/SNPD2007.pdf

[2] S. Lee and Y. Kim, "Analysis of the Resource Chain Model: the New Peer-to-Peer Reputation-Based Trust Model," *CAINE2007*, USA, 2007. http://tiger.towson.edu/~slee5/TH_1641.pdf

[3] S. Lee, K. Park and Y. Kim, "The New P2P Reputation Model Based on Trust Management System," *UKC2007*, USA, 2007. http://tiger.towson.edu/~slee5/UKC2007.pdf

[4] K. Aberer and Z. Despotovic, "Managing Trust in a Peer-2-Peer Information System," *ACM CIKM*, USA, 2001. http://lsirpeople.epfl.ch/despotovic/CIKM2001-trust.pdf

[5] Y. Wang and J. Vassileva, "Trust and Reputation Model in Peer-to-Peer Networks," *Proc. of IEEE Conference on P2P Computing*, Linkoeping, Sweden, September 2003. http://julita.usask.ca/Texte/Yao&JulitaP2Pfinal.pdf

[6] V. Bhat, "Survey paper: Reputation Management in Peer-to-peer Systems," University of Texas at Austin, 2004. http://www.cs.utexas.edu/~vishwas/documents/Reputation.pdf

[7] E. Damiani, D. C. Vimercati, S. Paraboschi, "A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks,"

CCS'02 Washington DC, USA, 2002. http://gnunet.org/papers/p207-damiani.pdf

[8] M. Gupta, P. Judge, and M. Ammar, "A Reputation System for Peer-to-Peer Networks," *NOSSDAV'03*, USA, 2003. http://www.cs.indiana.edu/~minaxi/pubs/reputation.pdf

[9] S. Kamvar, M. Schlosser, and H. Garcia-Molina, "The EigenTrust Algorithm for Reputation Management in P2P Networks," *WWW2003*, May 2003. http://www.stanford.edu/~sdkamvar/papers/eigentrust.pdf

[10] S. Kamvar, M. Schlosser, and H. Garcia-Molina, "EigenRep: Reputation Management in P2P Networks," *WWW2003*, Hungary, 2003. http://www-sccm.stanford.edu/pub/sccm/sccm02-16.pdf.old

[11] D. Milojicic, "Peer-to-Peer Computing," HP Laboratories, Palo Alto, July. 2003. http://www.hpl.hp.com/techreports/2002/HPL-2002-57R1.pdf

[12] A. Oram, "Peer-to-Peer Harnessing the Power of Disruptive Technologies," Oreilly, 2001, pp 45-49.

[13] A. Selcuk, E. Uzun, M. R. Pariente, "A Reputation-Based Trust Management System for P2P Networks," *International Journal of Network Security*, Vol.6, No.2, PP.227-237, Mar. 2008. http://www.ics.uci.edu/~euzun/pub/ijns-2008-v6-n3-p235-245.pdf

[14] V. Jumppanen, "File reputation in decentralized P2P reputation management," *HUT, Seminar in Internetworking*, 2005. http://www.tml.tkk.fi/Publications/C/18/jumppanen.pdf

[15] L. Xiong and L. Liu, "PeerTrust: Supporting Reputation-Based Trust in Peer-to-Peer Communities," *IEEE Transactions on Knowledge and Data Engineering (TKDE),* Special Issue on Peer-to-Peer Based Data Management, 16(7), PP.843-857, July 2004. http://www.mathcs.emory.edu/~lxiong/research/pub/xiong04peertrust.pdf

[16] L. Xiong and L. Liu, "A Reputation-Based Trust Model for Peer-to-Peer eCommerce Communities," *In IEEE Conference on Electronic Commerce (CEC'03),* Newport Beach, June 2003. http://www-static.cc.gatech.edu/~lingliu/papers/2003/xiong03reputation.pdf

[17] K. Aberer, "P-Grid: A Self-organizing Structured P2P System," *SIGMOD Record*, Sep. 2003. http://lsirpeople.epfl.ch/aberer/GMD-PAPERS/Sigmod%20Record%20P2P.pdf

[18] S. Marti, "Limited Reputation Sharing in P2P Systems," *EC'04*, New York, USA, 2004. http://delivery.acm.org/10.1145/990000/988787/p91-marti.pdf?key1=988787&key2=4550503021&coll=portal&dl=ACM&CFID=565551514&CFTOKEN=565551514

[19] L. Liu, K. Ryu, and Partha Dasgupta, "R-Chain: A Self-Maintained Reputation Management System in P2P Networks," *ISCApdcs,* 2004, pp. 131-136. http://cactus.eas.asu.edu/partha/Papers-PDF/2004/PDCS-Lintao.pdf

[20] M. MacDonald, "Peer-to-peer with VB.NET," APress, 2003.

[21] D. Wallach, "A Survey of Peer-to-Peer Security Issues," *International Symposium on Software Security*, Tokyo, Japan, November 2002.

http://www.eecs.harvard.edu/~mema/courses/cs264/papers/securitySurvey-swSecurity2002.pdf

[22] R. Subramanian and B. Goodman, "Peer-to-peer computing: the evolution of a disruptive technology," Idea Group Publishing, 2005, pp 28 – 65.

[23] A. Jacklin, "Using UDP to increase the Scalability of Peer-to-Peer networks," Master's thesis, The University of South Wales, Sydney, Australia, 2003. http://www.cse.unsw.edu.au/~leozc/paper/u9aj.pdf

[24] S. Lee, "A Trust Management Scheme in Structured P2P Systems," *AP2PC*, 2005.

[25] W. Yang and N. Abu-Ghazaleh, "GPS: A General Peer-to-Peer Simulator and its Use for Modeling BitTorrent," *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005*, pp. 425-432. http://www.cs.binghamton.edu/~wyang/gps/mascots05.pdf

[26] S. Naicken, A. Basu, B. Livingston and S. Rodhetbhai, "A Survey of Peer-to-Peer Network Simulators," *Proceedings of the Seventh Annual Postgraduate Symposium*, Liverpool, UK, 2006. http://www.cms.livjm.ac.uk/pgnet2006/Programme/Papers/2006-057.pdf

[27] A. Singla and C. Rohrs, "Ultrapeers: Another Step towards Gnutella Scalability," *Lime Wire LLC*, 2001. *http://rfc-gnutella.sourceforge.net/Proposals/Ultrapeer/Ultrapeers.htm*

[28] M. Ham and G. Agha, "ARA: A Robust Audit to Prevent Free-Riding in P2P Networks," *In: Proceedings of the Fifth IEEE International*

Conference on Peer-to-Peer Computing, 2005, pp. 125–132. http://osl.cs.uiuc.edu/docs/P2P2005ARA/ham-ARA.pdf

[29] E. Adar and B. Huberman, "Free riding on Gnutella," Tech. Rep.,HP, 2000. http://citeseer.ist.psu.edu/cache/papers/cs/15622/http:zSzzSzwww.parc.xerox.comzSzistlzSzgroupszSzieazSzpaperszSzgnutellazSzGnutella.pdf/adar00free.pdf

[30] X. Jin, S. Chan, W. Yiu, Y. Xiong and Q. Zhang, "Detecting malicious hosts in the presence of lying hosts in peer-to-peer streaming," in *Proceedings of IEEE International Conference on Multimedia Expo (ICME)*, Toronto, Canada, 9-12 July 2006, pp. 1537-40. http://ihome.ust.hk/~kenyiu/pub/ICME06_MDA.pdf

[31] A. Singh, M. Castro, P. Druschel, and A. Rowstron, "Defending against Eclipse attacks on overlay networks," *in Proc. SIGOPS EW'04*, 2004. http://www.cs.rice.edu/~druschel/publications/Eclipse-EW.pdf

[32] S. Park, L. Liu, C. Pu, M. Srivasta, and J. Zhang, "Resilient Trust Management for Web Service Integration," *Proceedings of the 3rd IEEE International Conference on Web Services, ICWS*, 2005. http://www-static.cc.gatech.edu/~lingliu/papers/2005/icws-trust-Park.pdf

[33] R. Gupta and A. Somani, "Reputation Management Framework and its use as Currency in Large-Scale Peer-to-Peer Networks," *Fourth International Conference on Peer-to-Peer Computing (P2P'04,)* 2004. http://csdl.computer.org/dl/proceedings/p2p/2004/2156/00/21560124.pdf

[34] M. Haridasan and R. Renesse, "Defense against Intrusion in a Live Streaming Multicast System," *Proceedings of the 6th IEEE International Conference on Peer-to-Peer Computing (P2P2006)*, September, 2006. http://www.cs.cornell.edu/projects/quicksilver/public_pdfs/secureStreamPaper.pdf

[35] S. Marti, H. Garcia-Molina, "Identity Crisis: Anonymity vs. Reputation in P2P Systems," *Third International Conference on Peer-to-Peer Computing (P2P'03)*, Sweden, 2003. http://www.eecs.umich.edu/~zmao/eecs589/papers/Marti04.pdf

[36] N. Stakhanova, S. Ferrero, J. Wong, and Y. Cai, "A Reputation-based Trust Management in Peer-to-peer Network Systems," *2004 International Workshop on Security in Parallel and Distributed Systems In Proceedings of 17th International Conference on Parallel and Distributed Computing Systems, PDCS 2004.* http://www.cs.iastate.edu/~ndubrov/201.pdf

[37] General Purpose Simulator for P2P network http://www.cs.binghamton.edu/~wyang/gps/

[38] P2PSIM: a simulator for peer-to-peer protocols http://pdos.csail.mit.edu/p2psim/faq.html

[39] P. Dewan and P. Dasgupta, "Trusting Routers and Relays in Ad hoc Networks," *First International Workshop on Wireless Security and Privacy (WiSPr 2003), (in conjunction with IEEE ICPP 2003),* Kahosiung, Taiwan, October 2003. http://cactus.eas.asu.edu/partha/Papers-PDF/2003/Dewan-Wispr.pdf

[40] The FreeNet Homepage http://freenet.sourceforge.net/

[41] The Gnutella Homepage http://gnutella.wego.com/

[42] The Gnutella Developer Homepage http://gnutelladev.wego.com/

[43] Q. Zhang, T. Yu and Keith Irwin, "A Classification Scheme for Trust Functions in Reputation-Based Trust Management," *A workshop at ISWC*, Hiroshima, Japan, 2004. http://trust.mindswap.org/trustWorkshop/papers/4-f.pdf

Curriculum Vitae

Sinjae Lee Curriculum Vitae

Department of Computer and Information Sciences
Towson University
8000 York Road
Towson, Maryland 21252
USA

Voice: (410)704-2428
Email: slee5@towson.edu

Education

2008	*Doctor of Science Applied Information Technology,* Towson University, Towson, MD
2005	*Doctor of Engineering Computer Engineering (In Progress)* Cleveland State University, Cleveland, OH
2001	*Master of Science Computer Science* Towson University, Towson, MD
1999	*Bachelor of Science Computer Science* Towson University, Towson, MD
1990	*Bachelor of Science Mathematical Education* Kookmin University, Seoul, South Korea

---

Research Interests

- Network Security
- Peer-to-peer Reputation Systems
- Network Simulation
- Wireless Network

Publications, Presentations, Paper and Project

[1] Sinjae Lee, Shaojian Zhou, and Yanggon Kim, "Analysis on Malicious Peers' Behaviors of P2P Trust Model: The Resource Chain Model," *SNPD2008*, Phuket, Thailand, 2008.

[2] Sinjae Lee and Yanggon Kim, "Analysis of the Resource Chain Model: the New Peer-to-Peer Reputation-Based Trust Model," *CAINE2007*, San Francisco, USA, 2007. http://tiger.towson.edu/~slee5/TH_1641.pdf

[3] Sinjae Lee, Shaojian Zhou, and Yanggon Kim, "P2P Trust Model: The Resource Chain Model," *SNPD2007*, Qingdao, China, 2007. http://tiger.towson.edu/~slee5/SNPD2007.pdf

[4] Sinjae Lee, Kyungeun Park, and Yanggon Kim, "The New P2P Reputation Model Based on Trust Management System," *UKC2007*, Virginia, USA, 2007. http://tiger.towson.edu/~slee5/UKC2007.pdf

[5] Kyungeun Park, Sinjae Lee, Juno Chang, and Yanggon Kim, "Design of Massive Events Stream Service Architecture." *UKC2007*, Virginia, USA, 2007. http://tiger.towson.edu/~slee5/UKC2007_MESSA.pdf

[6] Sinjae Lee, "Survey Paper: Implementation of Software Radio with Wireless Networking," Cleveland State University, Cleveland, OH, 2003. http://tiger.towson.edu/~slee5/eec793_paper.pdf

[7] Sinjae Lee, "Final Project Report: An Online Outpatient Reservation System," Towson University, Towson, MD, 2001. http://tiger.towson.edu/~slee5/gradproj.pdf

---

Research Experience

2005-present

Doctoral Research: Approach to the New Peer-to-Peer Reputation- based Trust Model
Department of Computer and Information Science
Towson University, Towson, MD
Advisor: Yanggon Kim Ph.D.

This dissertation proposes the new P2P reputation-based trust model. The main idea of the model is using a routing table to record the information of nodes' credibility and their recommending nodes' credibility with feedback. From the test result, compared with the system without the reputation control, the self-adjusting system can help the system itself in

automatically decreasing the impact of the malicious users in this group. Such a basic idea of the model can serve the P2P file sharing system well. Therefore, using this model can help us efficiently find the best and safest resource location and decrease the number of malicious transactions.

2002-2005

Doctoral Research: Implementation of Software Radio with Wireless Networking, Energy efficiency with software radio, Multimedia benchmarking
Mobile Computing Research Laboratory
Department of Electrical and Computer Engineering
Cleveland State University, Cleveland, OH
Advisor: Chansu Yu Ph.D.

Software Radio is a wireless communication that enables seamless connectivity in any place, any time and which provides flexibility of choosing from the available communication standards. There are many wireless communication standards such as DAB, DVB, cellular GSM, Wireless LAN, etc. They cannot operate in a mobile device because they have different frequency bands. Therefore, a mobile device is needed for using the different frequency bands. The mobile device can be Software Radio. This research presents the implementation of software radio systems in the flexibility. Low power issue is presented.

1999-2001

Master's Research: An Online Outpatient Reservation System
Department of Computer and Information Science
Towson University, Towson, MD
Advisor: Yanggon Kim Ph.D.

The current OORS is a web-based, three-tier, client/server application system designed to allow outpatients, doctor's secretaries, and administrators to finish their jobs online and tries to replace the conventional outpatient reservation. Public interesting in this is rising on how to make a Web interface more efficiently and conveniently. In this research, the Web interface at the first tier provides a user-friendly and more convenient way than the current OORS to allow the client passing the data to the Web server. The current OORS has two main parts, which are for outpatient and for the employee. This project will focus on outpatient and the outpatient can make a new appointment with a doctor more efficiently. As the result, this project will provide a more convenient GUI than the current OORS. The OORS could be used in the real world such as in a hospital or doctor's

office and could be a Web-based solution to conventional outpatient reservation.

Teaching Experience

2007-present
Lecturer
Department of Computer and Information Science
Towson University
Towson, MD

Course developed & taught
Computer and Creativity (COSC 109)
Assisted
A Department Faculty

2005-2006
Teaching Assistant
Department of Computer and Information Science
Towson University
Towson, MD

Course taught
Computer and Creativity (COSC109)
Assisted
A Department Faculty

2004
Teaching Assistant
Department of Electrical and Computer Engineering
Cleveland State University
Cleveland, OH

Assisted
A Department Faculty

Work Experience

1991-1993
Full-time Officer
Multimedia Materials' Factory
Korea Air Force
Chungwon, South Korea

Managed and Developed
To Produce Multimedia Materials

1990
Full-time Officer

A Training School of Korea Air Force
Korea Air Force
Sachun, South Korea

Taught and Developed
Basic Skills of Military Training

Skills

- Extensive experience with DOS, Windows computers, also familiar with UNIX environments
- Proficient in the following languages: Assembly, C, C++, LISP, SIMSCRIPT, Pascal, & JAVA
- Highly experienced in management information system & data analysis
- Skilled in foreign languages
- Speaks and reads Korean fluently

Awards

- Graduate Student Scholarship Award (2005-2008) - Full-time Graduate Assistant
- Graduate Student Association Award (2007)
- Undergraduate Student Scholarship Award (1988)

License

- A Middle or High School Teacher's License in Korea (1990)
