ABSTRACT

Title of dissertation: ROBUST TRUST ESTABLISHMENT IN DECENTRALIZED NETWORKS

Chuk-Yang Seng, Doctor of Philosophy, 2010

Dissertation directed by: Associate Professor William Arbaugh
Professor Ashok Agrawala
Department of Computer Science

The advancement in networking technologies creates new opportunities for computer users to communicate and interact with one another. Very often, these interacting parties are strangers. A relevant concern for a user is whether to trust the other party in an interaction, especially if there are risks associated with the interaction.

Reputation systems are proposed as a method to establish trust among strangers. In a reputation system, a user who exhibits good behavior continuously can build a good reputation. On the other hand, a user who exhibits malicious behavior will have a poor reputation. Trust can then be established based on the reputation ratings of a user. While many research efforts have demonstrated the effectiveness of reputation systems in various situations, the security of reputation systems is not well understood within the research community. In the context of trust establishment, the goal of an adversary is to gain trust. An adversary can appear to be trustworthy within a reputation system if the adversary has a good reputation.

Unfortunately, there are plenty of methods that an adversary can use to achieve a good reputation. To make things worse, there may be ways for an attacker to gain an advantage that may not be known yet. As a result, understanding an adversary is a challenging problem. The difficulty of this problem can be witnessed by how researchers attempt to prove the security of their reputation systems. Most prove security by using simulations to demonstrate that their solutions are resilient to specific attacks. Unfortunately, they do not justify their choices of the attack scenarios, and more importantly, they do not demonstrate that their choices are sufficient to claim that their solutions are secure.

In this dissertation, I focus on addressing the security of reputation systems in a decentralized Peer-to-Peer (P2P) network. To understand the problem, I define an abstract model for trust establishment. The model consists of several layers. Each layer corresponds to a component of trust establishment. This model serves as a common point of reference for defining security. The model can also be used as a framework for designing and implementing trust establishment methods. The modular design of the model can also allow existing methods to inter-operate.

To address the security issues, I first provide the definition of security for trust establishment. Security is defined as a measure of robustness. Using this definition, I provide analytical techniques for examining the robustness of trust establishment methods. In particular, I show that in general, most reputation systems are not robust. The analytical results lead to a better understanding of the capabilities of the adversaries. Based on this understanding, I design a solution that improves the robustness of reputation systems by using accountability. The purpose of account-

ability is to encourage peers to behave responsibly as well as to provide disincentive for malicious behavior.

The effectiveness of the solution is validated by using simulations. While simulations are commonly used by other research efforts to validate their trust establishment methods, their choices of simulation scenarios seem to be chosen in an ad hoc manner. In fact, many of these works do not justify their choices of simulation scenarios, and neither do they show that their choices are adequate. In this dissertation, the simulation scenarios are chosen based on the capabilities of the adversaries. The simulation results show that under certain conditions, accountability can improve the robustness of reputation systems.

ROBUST TRUST ESTABLISHMENT IN DECENTRALIZED NETWORKS

by

Chuk-Yang Seng

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2010

Advisory Committee:

Associate Professor William Arbaugh, Chair/Advisor
Professor Ashok Agrawala, Co-advisor
Associate Professor Atif Memon
Assistant Professor Jennifer Golbeck
Associate Professor Michel Cukier, Dean's Representative

To my wife

ACKNOWLEDGMENTS

I am thankful to my advisor, Associate Professor William Arbaugh, for the years of support and the freedom to pursue my research interests. He was patient with me during the times of research difficulty, as well as supportive of my personal situations. I enjoyed the numerous discussions we had throughout the years. These discussions guided me to the finding of this research topic and the completion of this dissertation. I wish him all the best in his career.

I would also like to express my appreciation to Professor Ashok Agrawala, my co-advisor, for stepping in and for providing advice during Professor Arbaugh's leave of absence.

I would like to extend my gratitude to my committee members for their precious time, challenges and advice: Assistant Professor Michel Cukier (Dean's Representative), Assistant Professor Jennifer Golbeck and Associate Professor Atif Memon. I would also like to thank Professor James Hendler, who was in my proposal committee, for providing valuable insights to the research topic. It is unfortunate that he left the University of Maryland and was unable to participate in my defense committee.

During this period, I have learned much from my friends at the Maryland Information and Systems Security Lab (MISSL). I enjoyed the various weekly discussions with Nick Petroni, Tim Fraser, T. Charles Clancy, Arunesh Mishra, Minho Shin and Yuan Yuan. In particular, I learned a lot from Tim Fraser while working on iButton and trusted computing research.

I would also like to thank my English editor, Ms Sharon vonBergener, from the English Editing for International Graduate Students program at the University of Maryland.

Finally, none of this would have been possible without the support of my family members. I am especially grateful to my wife, Su-Eng Tan, for her love, companionship and putting up with me during this period of time.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The term, "trust management", was coined by Blaze in [12] to refer to the problems of formulating security policies, security credentials, verifying whether particular sets of credentials satisfy the relevant policies, and deferring trust to third parties. Policies define sets of conditions in which trust can be established. These conditions are typically expressed in policy languages, such as PolicyMaker [12] and KeyNote [11]. A principal can gain trust by satisfying the policies. Satisfaction of policies is usually achieved by presenting the required credentials and by proving ownership of the credentials to the policy verifier. To prevent mistrust, it is important that a principal is not able to obtain any credentials that he is not entitled to. Despite the importance of credentials attainment, it is seldom addressed in trust management research. In fact, credential attainment is not listed as a component of trust management by Blaze. This is because the domain of trust management is often applied to organizations where credentials are created and distributed offline. For example, an organization can be described by a hierarchy of roles. These roles represent the authorities within an organization. The role of a member within the organization serves as a member's credential by defining what a member is authorized or not authorized to do. In such organizations, trust is closely related to authority.

A member with authority over some actions is said to be trusted on those actions. To illustrate, consider a bank, where an accounts manager may be authorized to approve loans below a certain amount, say, $2000. A trust management policy language can describe this policy by asserting that for a request to transfer any amount that is less than $2000 to a client, then the request must be digitally signed by any bank employee whose role is accounts manager. This essentially translates the problem of trust establishment into the problem of authentication. The policy verifier must authenticate that the request originates from an accounts manager. Authentication within an organization can be easily addressed by the use of public key cryptography, such as RSA [60], and key management solutions [36, 83, 23]. For example, by using X.509 public key certificates[36] to bind public keys to identities, one can use public key cryptographic techniques to prove his identity by demonstrating his knowledge of the private key that corresponds to the public key in the certificate.

With the proliferation of the Internet, many computer users are able to interact with one another over the Internet. Very often, these interactions involve strangers. When strangers interact, a valid question to ask is how can one user trust another user. This is especially crucial if the misplacement of trust results in substantial losses. For example, in an online auction, the act of bidding reflects the act of placing trust on the seller to deliver the goods after collecting the payment. If a bidder chooses to buy from a malicious seller, the seller may not deliver after collecting the payment, resulting in monetary loss for the bidder. Since the Internet is an open environment where almost anyone can participate, establishing trust

on the Internet is a greater challenge as compared to establishing trust within an organization. The first challenge is the value of identity. An identity within an organization can be mapped into a role and its associated authority within the organization. The role and authority information can decide the set of actions that a principal is trusted to perform. On the Internet, the users are usually free to choose any pseudonym to represent themselves. Pseudonyms are problematic [25] as they do not convey information that can be used to judge the trustworthiness of principals behind the pseudonyms. Moreover, pseudonyms are not secure since anyone can pick any pseudonym to represent himself and can easily change pseudonyms when things go bad.

The second challenge is the security of earning credentials. Within an organization, credentials are obtained offline. For example, employees earn credentials through actual job performances that lead to promotions. On the Internet, a common credential is reputation. Popular Internet applications, such as e-commerce applications (eBay [1], Amazon[2]), discussion sites (Stack Overflow[3]), product reviews (Epinions[4]) and crowd sourcing application (Mechanical Turk[5]), rate their participants based on their behaviors. Reputation serves as a way to judge the trustworthiness of a principal locally, based on the principal's past history of online activities within an Internet application. While the use of online activities to establish credentials mimics how credentials are earned within organizations, additional

[1] www.ebay.com

[2] www.amazon.com

[3] www.stackoverflow.com

[4] www.epinions.com

[5] www.mturk.com

security measures are essential to ensure that adversaries cannot manipulate various computing resources so as to inflate their reputation artificially.

The third challenge is due to the lack of centralized administrative supports on some Internet applications. These are typically applications that are built on decentralized and self-organized architectures, such as Peer-to-Peer (P2P) networks. Within an organization, a principal may abuse his authority by performing actions that bring harm to the organization (commonly known as the Byzantine Generals problem [42]). There are measures that an organization can fall back on should such events happen. For example, organizations can make use of audit trails and other legal means to protect themselves. However, such measures require some administrative and enforcement efforts which may not be present on decentralized Internet applications. For example, the use of pseudonyms means that to prosecute a malicious user, there must be some administrative efforts to link pseudonyms to their corresponding physical identities. Given the interesting challenges of decentralized environments, this dissertation is devoted to addressing the security of trust establishment in decentralized environments.

With the shift of computing interests towards the Internet, the traditional means of establishing trust for an organization through trust management are no longer sufficient. While various methods to establish trust [4, 59, 28] have been used, the security consequences are not well understood, as explained later. This dissertation attempts to address the security issues of trust establishment by defining and improving security in the context of trust establishment.

1.1 Research Problems

The main theme of this dissertation is to address the security problems arising from trust establishment. The concept of security is not well understood within the trust research community. This can be observed by the interpretation of security in different papers in the literature. For example, security is interpreted as authentication in [9, 19, 79], preventing collusion among malicious participants in [44, 40], maintaining privacy in [55], access control in [72], preventing Sybil attacks [22, 64], and as a probabilistic threat model in [6, 78]. Since security itself is a complex topic, numerous differing interpretations among various research efforts are understandable. However, different opinions mean that different research efforts focus only on a subset of security problems. This leads to concerns about how a particular trust establishment method will perform when attacked by a method that has not been considered by the researcher. For example, how will the method in [19] react to collusion attacks described in [40]? This poses the following relevant research questions:

- What are the semantics of security in the context of trust establishment?
- How does one determine whether a trust establishment method is secure?
- Is an existing trust establishment method secure against unanticipated attacks?

In addition, research efforts that focus on trust establishments are typically concerned with the aspects of computing and propagating trust [3, 4, 21, 28, 29, 32,

47, 49]. While these research efforts address important issues of trust establishment, security is not the primary focus. As such, if someone wishes to apply these methods in a practical setting, how does one go about adding security to these existing methods?

Reputation is a popular method for establishing trust in decentralized environments. This popularity is due to the fact that reputation systems can be set up without the need of centralized administration [19, 40, 72, 5]. Since this dissertation focuses on trust establishment in decentralized environments, the use of reputation as a means of establishing trust is of great interest to this research in particular, how secure are reputation systems. Reputation systems pose some interesting security problems, often involving the interference of human actions. For example, some security problems are mentioned in [59], which include inaccurate reputation, as well as malicious users masquerading as honest users by behaving well to establish good reputations prior to their malicious activities. In addition, the challenges of trust establishment presented in the previous section adds to the security concerns of reputation systems.

Besides security, trust establishment itself is a complex topic, with a number of different methods for establishing trust described in the survey [31]. Most research efforts only focus on one aspect of trust establishment. For example, research on reputation systems [59] focuses solely on using reputation for trust establishment and nothing else. Given that different trust establishment methods have different strengths and weaknesses, designing a trust establishment solution based on the composition of various methods may lead to a better solution since different methods

may complement each other. For example, one problem with reputation systems is the honesty of the people providing feedback [58]. Therefore, combining reputation systems with a solution that promotes honesty and discourages dishonesty may lead to better results. This leads to the problem of how one can design a system that allows different methods to be composed.

1.2 Contributions

This dissertation addresses the research problems mentioned in the previous section. By seeking answers to the above problems, this dissertation contributes towards better understanding of trust establishment and security. In the area of secure trust establishment, the contributions include precise definition of security, as well as techniques for identifying and solving security problems. The contributions can be listed in detail as follows:

1. Model for trust establishment. Before exploring the meaning of security in the context of trust establishment, it is necessary to have a common understanding of trust establishment. With numerous research efforts, such as [47, 4, 40], focusing on specific computation aspects of trust establishment, the model defines a generic framework for trust establishment. More specifically, the trust establishment model abstracts the process of trust establishment by breaking it down into simpler components. The components are stacked up to form a layered model. Each layer interacts with the adjacent layers by providing services to the upper layer and consuming services from the bottom layer.

This model is similar to the OSI model for networking which abstracts the networking process into seven layers, where the specifics of each layer are left to the individual protocol designers and implementers. Taking the same abstract approach to trust establishment has its advantages. First, the model captures the essential components of trust establishment, rather than focusing on specific aspects of trust establishment. This provides a common definition when trust establishment is discussed. Second, the model provides a context in which security can be defined. Third, just like how the OSI networking model provides a framework for developers to implement networking solutions, the trust establishment model provides the framework for implementing trust establishment solutions, including the ability to compose various specific methods together.

2. Security definition. Security for trust establishment can be defined on two levels. The first level defines security in the authentication sense. To establish trust in a principal, the principal and the credentials owned by the principal must first be identified. The second level defines security in the semantic sense. Given that the purpose of trust establishment is to distinguish between honest and malicious principals, a trust establishment method is said to be secure if it is capable of performing this task with a high rate of accuracy. The first definition of security in terms of authentication has already been well researched in the domain of systems and network security, resulting in solutions for authentication, such as X.509 [36] and Pretty Good Privacy (PGP) [83],

as well as credentials management, such as SDSI/SPKI [23]. Therefore, the contribution of this dissertation is more towards the semantic sense of security. To achieve this, this dissertation describes a formal approach to define security. This security definition is then used to evaluate trust establishment solutions to determine whether a particular solution is sufficiently secure.

3. Adversarial analysis. Security is informally described as the ability of a trust establishment method to distinguish between honest and malicious principals. The goal of a malicious principal is to make it difficult for a trust establishment method to distinguish himself from an honest principal. Adversarial analysis plays an important role in analyzing the security of a trust establishment method. Adversarial analysis attempts to discover ways in which an adversary can gain trust successfully. This dissertation presents an analytical method, ExBAN, as a technique for adversarial analysis. ExBAN is an extension of BAN logic [14]. BAN is a form of belief logic that is successfully used in analyzing authentication protocols for loopholes. Unlike those previously mentioned trust establishment research that define security based on specific forms of attacks, ExBan analyzes security based on the flow of information. This is because trust decisions are dependent upon the accuracy of available information. An adversary can gain trust if the information leads the trust decision maker to believe that the adversary is honest. Given an information flow description of a trust establishment process, ExBAN provides a set of inference rules to establish beliefs in the accuracy of the information. Ana-

lyzing security based on the information flow model has the advantage that it is independent of specific attacks. For example, in reputation systems, an adversary may have a good reputation (which is one of the information used for establishing trust) by behaving well for some time, or by colluding with other adversaries as described in [40]. Although the attack methods may be different, the results are the same that is, the reputation of an adversary is artificially inflated. From the point of view of information flow analysis, these attacks belong to the same class since they affect the same piece of information and they result in the same consequence. If a new type of attack surfaces in the future, the security properties of a trust establishment method will be unaffected as long as the new attack belongs to a known class.

4. Improving the security of specific instances of reputation systems in decentralized P2P networks. Decentralized networks are chosen as the trust establishment environment since they add to the challenge of securing reputation systems, due to the absence of centralized authorities. Since reputation systems are commonly used, the dissertation focuses on improving the security of trust establishment as well as to serve as a proof of concept for the above mentioned contributions. Using ExBAN, the security problems with reputation systems are identified. Using the trust establishment model as the framework, a solution is designed to mitigate the security problems. The solution is to supplement reputation systems with accountability. The motivation of using accountability is simple: the nature of an adversary is such that it does

not want to be held accountable for its actions. Therefore, an adversary can be distinguished from an honest principal by the willingness of a principal to establish accountability. Accountability can be established through different means, such as trading some form of privacy in order to gain trust. Simulations are performed to verify the effectiveness of using accountability to supplement reputation systems.

1.3 Organization

This dissertation is organized as follows. Chapter 2 will provide an overview of trust from various disciplines, as well as trust as a computational concept. This includes research that addresses the computation, establishment and propagation of trust. These will provide the necessary background for references to trust in this dissertation. Since this dissertation also focuses on trust establishment using reputation, Chapter 3 is devoted to the topic of the reputation system. In Chapter 3, I review various existing reputation systems in decentralized networks and their design features. The chapter will also critique these reputation systems, in particular, on security and validation (simulations) issues.

Chapter 4 defines trust establishment by presenting a layered model. The chapter describes the details of each individual layer and how different existing methods fit into the model. The model can also serve as an implementation framework in which existing methods can be composed to form a solution for trust establishment. The model provides an insight into how security can be defined in the context of

trust establishment.

Security is always a concern in this dissertation. The concept of security is first defined in Chapter 5. The chapter then presents ExBAN for analyzing the security of a trust establishment method. The results of the analysis lead to the understanding of how an adversary can gain trust. The focus is given to reputation systems in decentralized networks, which will serve as a proof of concept.

With an understanding of an adversary's capability in a reputation system, Chapter 6 presents the use of accountability to improve the security of reputation systems. The chapter will discuss how to establish accountability in decentralized networks. To validate the effectiveness of using accountability for trust establishment, simulations are performed to determine how the method works under different scenarios. The simulations scenarios are based on the capabilities of an adversary. The simulations demonstrate that the use of accountability can improve the security of reputation systems and are presented in Chapter 7. In addition, the simulations identify the conditions where the combination of reputation systems and accountability can achieve good performance with respect to security.

While this dissertation provides a new direction in the research of trust with a focus on security, it opens up other research opportunities. Chapter 8 discusses some potential for future research.

Finally, Chapter 9 concludes this dissertation, summarizing the research problems, the solutions and the validation of the solutions.

Chapter 2

Trust

2.1 Introduction

The study of trust spans many different fields, including sociology, psychology, philosophy, economics and computer science. This is not surprising, given that trust is a prominent feature of our everyday lives. This chapter reviews the various work on trust. The objective is to provide an understanding on the topic of trust from different disciplines so as to identify the important features of trust establishment.

2.2 The Meaning of Trust

Trust plays an important role in our daily lives. Therefore, the study of trust is not restricted to the field of Computer Science. This section presents some of the more influential works on trust.

2.2.1 Trust as Decision Making Under Risk

One of the earliest work on trust is by Deutsch [20]. In the words of Deutsch, trust is a form of decision making under risk:

1. *the individual is confronted with an ambiguous path, a path that can lead to an event perceived to be beneficial (Va^+) or to an event perceived to be harmful (Va^-);*

2. *he perceives that the occurrence of Va^+ or Va^- is contingent on the behavior of another person; and*

3. *he perceives the strength of Va^- to be greater than the strength of $Va+$.*

If he chooses to take an ambiguous path with such properties, I shall say he makes a trusting choice; if he chooses not to take the path, he makes a distrustful choice.

When Deutsch stresses that the consequences of a negative outcome is stronger than those of a positive outcome, he is suggesting that the trust decision maker has to take considerable risk when deciding to trust. More importantly, this interpretation means that there are no incentives (where $Va+$ is of greater strength) to influence the decision maker to make a trusting choice. From the perspective of security, an interesting question is: What happens when $Va+$ has greater strength than $Va-$? If an individual can be nudged towards an ambiguous path through incentives, then an adversary can influence the decision making process by offering incentives. For example, in an online auction, if someone is in desperate need of an item, then the person may feel the urge to bid for the item, regardless of the seller's reputation, especially when there are only a handful of sellers offering the item or when the price is attractive. This view of trust provides insights to design

trust establishment methods. The establishment of trust involves risks. For a trust establishment method to be secure, the risks have to be reduced. One form of risks reduction is to establish trust based on factors which adversaries have little or no influence on.

2.2.2 Trust as a Means to Reduce Complexity

While Deutsch's model defined trust as an act of making a choice under risk, Luhmann views trust as a means to reduce the complexity of an environment [46]. From a sociological perspective, the society is very complex and a decision is influenced by too many variables so that it is difficult to predict the outcome. Trust provides a way to reduce the complexity of everyday life so that a society can function. For example, Luhmann argues that without trust, one would be unable to get out of bed in the morning because the risk of doing so is high. To understand Luhmann's argument, consider the possibilities that can happen should one get out of bed. Perhaps, the person may slip and fall, or should the person decide to step out of the house, he faces the risk of being run over by a car. There are too many other possibilities that it is not possible to be exhaustive. However, despite such risks, most people are not bothered by them. This is because the likelihood of such possibilities is too low to affect a person's decision to get out of bed. By getting out of bed, a person implicitly trusts that such bad things will not happen. For example, the person trusts the skills of other drivers, so that they will not run him down. In this case, trust reduces the complexity of making a decision.

This definition of trust is often used in security. For example, the issue of binding an identity to a public key is a complex problem since it is difficult to verify that a public key actually belongs to an individual. The complexity of this problem is reduced by using trust. In a centralized model, all principals trust a hierarchy of authorities who certify the identity and key bindings. Even in a decentralized model such as PGP, a web of trust is used to reduce the complexity of the binding issue. The problem of trust in a decentralized model is more interesting because in the absence of authorities, all individuals have to decide for themselves who to trust.

According to this aspect of trust, trust is often established in complex environments. To model trust as a computational model, it is necessary to abstract complex environments into simpler components. The abstractions have to be simple such that it is computationally feasible to compute trust. At the same time, care has to be taken to prevent over simplification to the extent that security problems are hidden by the abstractions. Therefore, designing trust establishment methods is a challenging task that requires the maintenance of a delicate balance between abstraction and security.

2.2.3 Trust as a Form of Expectation

Barber views trust as an aspect of social relationships and as some form of an expectation about the future [8]. Barber defines trust to exhibit:

1. expectation of the persistence and fulfillment of the natural and moral social orders,

2. expectation of "technically competent role performance",

3. expectation that partners in interaction will "carry out their fiduciary obligations and responsibilities, that is, their duties in certain situations to place others' interests before their own".

Barber's definition of trust is based on competence as evaluated by standards defined by the society. For example, a degree from an accredited university is accepted in many societies as a measure of competence in a specific field of study. There is an expectation that the degree holder is capable of demonstrating certain expertise.

In addition, Barber's definition also includes trust in authorities. Authorities are expected to carry out certain duties. For example, a law enforcer is empowered to uphold the laws and is expected to perform this duty faithfully. On the Internet, certificate authorities are trusted for certifying the binding between an identity and a public key. This trust can also be translated into a form of expectation, where the authorities are expected to discharge their duties, such that the bindings can be taken as the truth.

Reputation is often used as a measure of competence in our society. For example, a business that can demonstrate its competence to its customers, such as by offering good and reliable services, will result in having a good reputation. Therefore, when someone is in need of a particular service, the reputations of the various service providers are often solicited. This is also why businesses spend a lot of resources in building up their brand names. On the Internet, the use of reputation

is a popular way for trust establishment. This is especially so for decentralized applications where other forms of measures of competency are absent.

The definition of trust as a form of expectation relies upon the presence of institutionalized trust [38]. Institutionalized trust refers to the recognition of some members and organizations within a society as authorities on certain subjects and placing trust in the authorities. The trust in authorities can be extended to other members of a society when the authorities delegate trust to these members. For example, the trust in a certificate authority, CA, is delegated to the certificates issued by the certificate authority, as well as certificates issued by other certificates authorities whom CA trusts. The presence of institutionalized trust and the delegation of trust is important for establishing trust in a society. By trusting (institutionalized trust) a relatively small set of members of a society, the institutionalized trust can be propagated to other members of the societies, allowing one to trust a larger set of members of the society. However, a security problem with this model is it works only if the institutionalized trust is strong. For example, a corrupted authoritative figure may accept bribes not to discharge his duties faithfully. This not only results in misplacing trust in the corrupted authority, but also results in misplacing trust on other undesirable members of the society due to the delegation of trust. The use of reputation on the Internet can be vulnerable to this problem if online adversaries can manipulate their own reputations. As mentioned in Section 2.2.1, trust decisions have to be dependent on factors that are not within the control of the adversaries.

2.2.4 Trust as Probability

Gambetta [26] presents trust as a subjective probability that an agent can use to assess whether another agent will perform a particular action. This assessment is done before one can monitor such action, or one may not even have the capability to observe such action. Moreover, the assessment can affect an agent's own choice of action. Since trust is treated as a probability, Gambetta uses values in the range of 0 to 1 to represent trust.

The interpretation of trust as a probability provides a direct way to represent trust computationally. For example, works such as [6, 78] model trust as a probabilistic threat model. However, from the perspective of security, the use of probability can be problematic if an adversary can adjust his behavior such that the probability distribution function can assign a high probability (trust) to the adversary.

2.3 Computational Trust

Given the rich interest in trust in computer applications, an immediate problem is to define a model to represent trust as a computational concept. The previous section introduces different perspectives about trust. While these definitions vary, they are all valid forms of trust. In fact, these differing forms of trust may complement one another. For example, while Deutsch's definition deals with the act of making choices, it does not explain the motivations behind the choices. This may be explained by Barber's or Gambetta's definition, where the measure of competence

or subjective probability analysis may influence the decision making process. Given the potential synergy when different aspects of trust work together, a computational trust model should encompass these different aspects.

Perhaps, one of the most influential works in computational trust is Marsh's Ph.D dissertation [47]. Marsh's motivation is to develop a way for interacting agents to establish trust among themselves. To do so, Marsh considers different aspects of trust amongst research efforts mentioned previously. Marsh's representation of trust can be described as:

$$T_x(y, \alpha) = Utility(\alpha) \times Impt(\alpha) \times \widehat{T_x(y)}, \tag{2.1}$$

where α represents the situation, while x and y represent the trustor and the trustee respectively.

Marsh calls $T_x(y, \alpha)$ the situational trust, which is how much x trusts y in situation α. $Utility(\alpha)$ and $Impt(\alpha)$ measure the utility and importance of the situation α respectively, while $\widehat{T_x(y)}$ is an estimation of the trustworthiness of y. The utility and importance can be interpreted as a measure of $Va+$ and $Va-$ in Deutsch's work.

$\widehat{T_x(y)}$ is estimated by considering the previous actions of y in identical or similar situations. To do so, it is necessary to maintain knowledge about y's previous activities. Assuming that each of y's previous actions can be mapped into a trust rating in the range of $[-1, +1)$, Marsh suggests different ways to obtain $\widehat{T_x(y)}$, depending on x's disposition. If x is an optimist, x selects the maximum rating from the history as the trust estimate, and conversely, x selects the minimum rating

if x is a pessimist. A realist computes the average of the ratings as the trust estimate.

Situational trust is defined as the value of the trust estimate, weighted by the utility and importance of the situation. Utility is based on the expected utility theory in economics [73]. Marsh measures the utility of the situation as the mean of the utility of all possible outcomes. The utility of an outcome measures the satisfaction derived from the outcome. It can, for example, be based on the costs and benefits analysis for the outcome. Marsh uses values in the range of $[-1, +1]$ to represent utility. The importance of a situation ranges from $[0, +1]$. Marsh does not consider negative importance. In fact, situations of negative importance can be treated as the opposite of situations with positive importance. For example, $Impt(\neg\alpha) = 1$ represents the concept that situation α is not important.

Marsh explains the difference between utility and importance as the difference between objective and subjective measures. Utility is an objective measure, that can be computed or estimated, such as by using costs and benefits analysis. On the other hand, importance is a subjective judgment of the situation by an individual and may vary in two identical situations at different times. To illustrate, Marsh uses the example of playing a dice game. The utility of playing the game can be measured by the amount of prize money. The importance of playing may be low, especially if the odds are stacked against the player. However, if the player believes that the odds are in his flavor, perhaps by cheating, then the importance of playing increases while the utility remains the same. The role of importance in the definition of situational trust is to represent the fact that the subjective judgment of a principal may change, even when faced in identical situations. The addition of

importance to the rationality of utility provides the formalism with more prescriptive and descriptive power. From the perspective of security, this is a potential pitfall that an adversary can exploit.

Having established the situational trust, the next step is to decide whether trust can be established. Marsh computes a cooperation threshold to determine this. If the situational trust is above the threshold, trust is established. In the multi-agent model, the establishment of trust is defined by the cooperation behavior of the trustor, and hence the term "cooperation threshold". Marsh proposes two cooperation threshold models:

$$Threshold_x(\alpha) = \frac{Risk(\alpha)}{Competence_x(y,\alpha) + \widehat{T_x(y)}} \times Impt(\alpha), \tag{2.2}$$

and

$$Threshold_x(\alpha) = \frac{Risk(\alpha)}{(Competence_x(y,\alpha) + \widehat{T_x(y)}) \times Impt(\alpha)}. \tag{2.3}$$

The difference between these two models is the role of importance. Formula 2.2 models the situation where the more important the situation is, the more the amount of situational trust is required for cooperation to take place. Therefore, increasing the importance of the situation raises the cooperation threshold. On the other hand, Formula 2.3 models the situation where the more important the situation is, the more one needs to get it done. Therefore, in this case, increasing the importance of the situation lowers the cooperation threshold to facilitate cooperation. The second model is less attractive with respect to security since the importance of an event actually makes it easier for trust to be established.

The cooperation threshold models demonstrate an insight into trust establish-

ment. The threshold models show that trust establishment is achieved by maintaining a balance between risk, competence of y and situational trust in y. Situations with high risk require greater competence and/or situational trust to lower the cooperation threshold. Conversely, low risk situations are more tolerant to less competence and/or situational trust.

The values for both risk and competence are within the range $[0, 1]$. Evaluating these values depend on an agent's past experiences and knowledge. Marsh suggests different methods for measuring these values depending on the availability of experiences and knowledge. Clearly, learning is important to provide more accurate measures for similar situations in the future.

While Marsh provides a thorough model for modeling trust establishment, the model itself is largely based on heuristics and upon careful study, reveals problems in some circumstances due to the heuristic nature of the model. An obvious problem is when $Competence_x(y, \alpha) = -\widehat{T_x(y)}$ in Formula 2.2. Marsh documents a list of problematic values in [47]. Despite these problems, the value of Marsh's model is that it encompasses the definition of trust from different fields of study. Marsh's model is also flexible, as witnessed by the use of different cooperation threshold models. As a matter of fact, since the model is based on heuristics, one can even design different heuristics for the various components of the model to adapt to different situations.

Besides Marsh's work, there are several research efforts in computational trust. In centralized environments where roles and authorities are clearly defined, the most common method is trust management [12]. Trust management involves the manage-

ment of digital credentials and policies. The main application of trust management is access control. The role of access control is to determine whether a subject x can access object y. The object y can be thought of as files with read, write or execute access rights. [1] To determine the access rights of x to y, x must have the necessary credentials as dictated by the access policies to y. The credentials of x may be defined by x's role within an organization, or by some digital artifacts issued by some authorities that attest to the capabilities of x, Examples of credentials include X.509 certificates [36] and PGP [83] for public key and identity binding, as well as SDSI/SPKI [23] for an authority to delegate responsibilities.

Credentials constitute one part of trust management. The other component of trust management is trust policies. Policy languages, such as PolicyMaker [12], KeyNote [11], TPL [37] and Referee [17] allows the definition of trust policies which specify the requirements, in terms of credentials, for trust to be established. The formulation of trust policies is very similar to access control. The role of access control is to determine the specific set of rights a principal has over an object. As a result, research in access control [33, 74] is applicable.

The work in trust management is extended to allow trust establishment across organizations. The namespace binding feature of SDSI/SPKI credentials allows one to identify the origin of the credentials. Trust establishment between two different organizations has been described in [79] where the types of inter-domain (organizations) trust relationships are explained.

[1] An action, such as the transfer of funds into an account can be thought of as executing a program that does the actual transfer.

To facilitate the satisfaction of trust policies between two organizations, trust negotiation [76, 75, 77] allows two parties to come into agreement on the set of credentials that are required for trust to be established.

In open, decentralized environments, papers such as [9, 44] deal with establishing trust in public keys, while works such as [3, 4, 21, 28, 29, 32, 49] suggest different approaches to compute trust in virtual communities. These works on trust establishment can be classified into two general approaches: reputation [59] and recommender [57] systems.

Reputation and recommender systems are popular methods for establishing trust. In reputation systems, each principal has a reputation that estimates the trustworthiness of the principal. The reputation of a principal is typically computed based on the principal's past behavior.

Recommender systems are often used in applications where participants can provide opinions on some subject, such as opinions about books, movies [29], restaurants and consumer products. These opinions serve as recommendations from the participants. In recommender systems, trust in a principal is determined by whether the principal's opinions are useful. Therefore, recommender systems establish trust between two principals by measuring the semantic distance between them. The semantic distance measures how similar these two principals are in terms of their opinions. Empirical results from [81, 82] have demonstrated that existence of correlation between trust and user similarity where opinions from principals with smaller semantic distances are more trusted.

Here I noted some of the popular research on trust establishment. A deeper

study of the operations of these trust establishment methods reveals that these methods can be abstracted into a two-step generic trust establishment framework. Using Marsh's terminology, the first step involves the computation of situational trust. The second step is to input the situational trust value into a decision function to derive an outcome. In trust management, the situational trust is defined in terms of credentials while the decision function is a policy checker that checks for the satisfaction of trust policies. For reputation and recommender systems, the situational trust is defined in terms of reputation ratings and semantic distances respectively. The decision making function for both systems are threshold functions, such as Marsh's threshold functions.

2.4 Trust properties and trust relationship

Trust establishment results in the formation of the trust relationship between at least two principals. Many research efforts have been devoted to the study of trust relationships to understand the properties of trust, to allow reasoning with trust and to propagate trust. Research on this topic leads to important results that allow trust to be established among strangers on the Internet and open networks.

Since different research efforts define trust differently, it is not a surprise that there is no universal agreement on a set of properties that trust relationships have to adhere to. The basic property of trust that is widely accepted is that trust is context sensitive. This is a simple property that can be observed from our daily lives. For example, one may trust his doctor on medical advice, but may not necessarily

trust the same doctor's advice on the stock market. It is also generally agreed that trust is not symmetric. That is, x's trust in y may not be the same as y's trust in x. For example, trust relationships between parents and children, or trust between managers and subordinates are clearly different.

An important aspect of trust establishment is the propagation of trust. This allows trust in a set of entities to be extended to a greater set of entities. The delegation of authorities mentioned previously is an example of trust propagation. Besides delegation, the property of transitivity is useful for propagating trust relationships. To illustrate, in Fig. 2.1, the solid arrows represent existing trust relationships between principals A, B, C and D. The labels on the arrows represent the trust level between 0 to 1 inclusively. Due to the transitivity property, trust can be established between A and C, and also between D and C, as shown by the dashed arrows. There is still a remaining problem of deciding the level of trust. An example is the "weighted" method: to compute the trust level between A and C, take the trust level between B and C, weighted by A's trust in B. As a result, A's trust in C is $0.7 \times 0.8 = 0.56$ and D's trust in C is $0.3 \times 0.8 = 0.24$.

Another relevant problem is how to aggregate trust values obtained from different paths. Consider the situation in Fig. 2.2. A can establish trust in C via two paths: $A \rightarrow B \rightarrow C$ and $A \rightarrow D \rightarrow C$. To obtain A's trust level in C, A can combine the trust rating from the two paths. For example, A may use the following:

$$T_{ik} = \frac{\sum_{j \in Adj(i)} T_{ij} \times T_{jk}}{\sum_{j \in Adj(i)} T_{ij}}, \tag{2.4}$$

where $Adj(i)$ is a set of nodes adjacent to i and T_{ij} is i's trust in j. Using Equa-

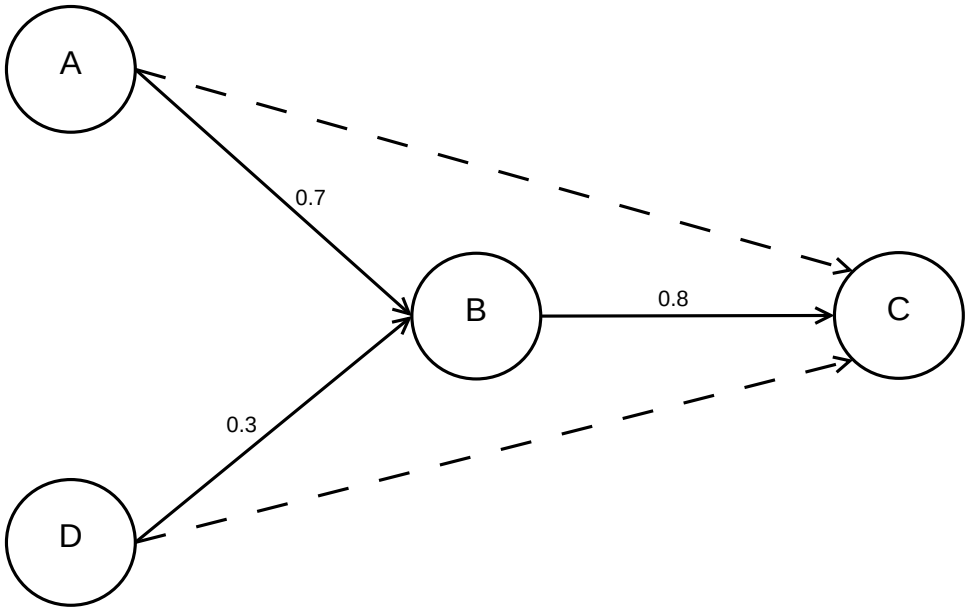

Figure 2.1: Propagation of trust using transitivity property.

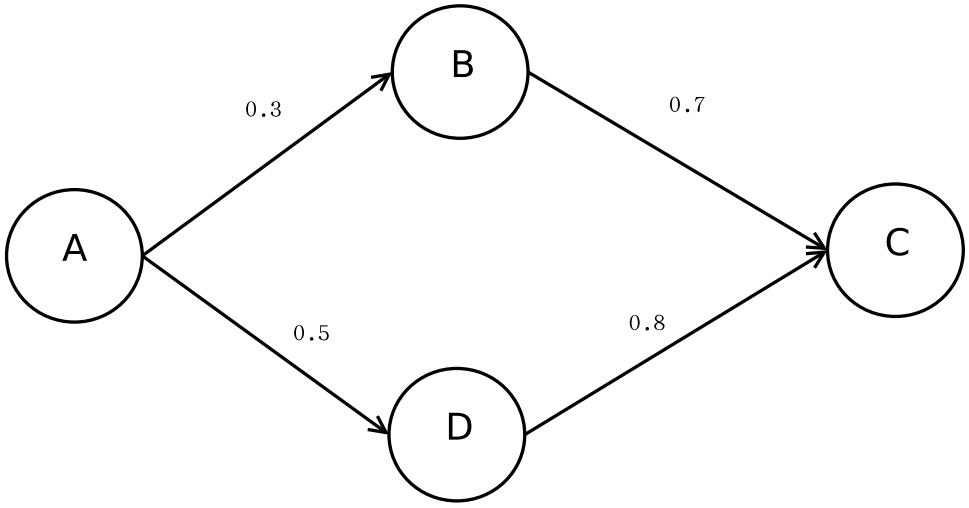

Figure 2.2: Aggregation of trust values from various trust paths.

tion 2.4, A's trust in C can be computed as:

$$\frac{0.3 \times 0.7 + 0.5 \times 0.8}{0.3 + 0.5} = 0.76$$

Despite the usefulness of the transitivity property, transitivity causes security risk [16]. If x trusts y and y trusts another principal z, then x also trusts z. Due to the transitivity property, y adds a trust relationship between x and z without x's explicit consent. Therefore, extra care must be taken when propagating trust through transitivity. In fact, to avoid security risks, trust should not be propagated via transitivity in most situations.

Recommendations [28, 80] is another method to propagate trust. If x trusts y to recommend a principal for context c, and y recommends z, then x trusts z for c. Strictly speaking, this is not a transitive trust relation, since two types of trust are involved: x's trust in y's ability to make recommendation in context c, and y's trust in z to perform in context c. In fact, this form of propagation is similar to delegation of trust. In the example, x delegates the task of making trust decision for c to y.

In [39], Jøsang differentiates the difference between trusting passionate entities with human-like capabilities and trusting rational entities that are basically systems. He defines trust as a belief that a rational entity will resist malicious manipulation or that a passionate entity will behave without malicious intent. While most work focuses on trust establishment among passionate entities, Jøsang's work provides an important insight that it is equally important to trust rational entities. This is evident in the prevalence of attacks that exploit vulnerabilities in software to gain control of computer systems. Therefore, when establishing trust, it is also important to verify that the computer system has not been compromised. Advances in Trusted Computing technologies [2, 61, 66] and hardware based monitor [56] provide the tools

to detect compromises in systems.

The security of trust propagation is addressed in some research, such as [44, 64]. If an adversary has established trust relationships with some principals, this trust can be propagated to other principals, resulting in other principals trusting the adversary. The authors of [44] design an algorithm that protects against keys that have been compromised. In [64], the scheme provides an interesting method to protect against the Sybil attack [22]. The Sybil attack is a form of attack when an adversary creates multiple identities.

2.5 The role of trust in this dissertation

While there are various definitions of trust, the intention of this dissertation is to be as generic as possible, such that the work can be applied to as many definitions of trust as possible. Therefore, it is intentional that the work in this dissertation does not provide a specific definition of trust. In general, trust can be defined as follows:

Definition 1 *Trust: Trust is a pairwise relationship between two principals, P and Q, such that if P trusts Q, then P expects Q to behave in a competent behavior over a period of time as defined by a specific context.*

The above definition captures the notion that trust is a relationship between two principals. This trust relationship describes a form of expectation that the trustee behaves in a competent manner. Competent behavior is defined by a specific context. Trust is also time sensitive. If P trusts Q at time t, it does not mean that

P will trust Q at some point in time other than t.

Since this dissertation uses trust in a computational setting, trust must be represented as a computational concept. An example of a computational representation of trust is Marsh's work, which in turn, incorporates the definitions of trust from various studies.

Marsh's computational model captures the essential steps for trust to be established. His model provides a basis for defining a trust establishment model in the later part of this dissertation. In addition, the properties and types of a trust relationship are relevant for analyzing the security of trust establishment.

Since this dissertation focuses on decentralized environments, the use of reputation as a computational trust establishment is of great interest. In the later chapters, reputation is used as an example, as well as the platform for performing experiments. Due to the interest in reputation, an entire chapter, Chapter 3, is allocated to discussing reputation. Despite the focus on reputation, the work presented in this dissertation is applicable to any computational trust representation.

Chapter 3

Reputation Systems

3.1 Introduction

Reputation systems [59] describe a class of trust establishment methods that primarily compute the reputation of a principal and utilize reputation to establish trust. The reputation of a principal is typically rated according to the principal's past behavior. In general, a principal with good behavior leads to the increment of the principal's reputation rating, while bad behavior leads to the decrement of the reputation rating. Over time, reputation provides a reflection of a principal's past history. A principal with a good reputation is more likely to be trusted since the principal has demonstrated good behavior in the past. Therefore, reputation can serve as an indication of how trustworthy the principal is.

According to [58], the reason why reputation works as a trust establishment method is due to the expectation of reciprocity and retaliation in future transactions. This means that a principal with a good reputation can expect to be rewarded in future interactions, such as receiving more cooperation from other principals. On the other hand, a principal with a bad reputation may be avoided by others in future. In fact, research [7, 35, 45, 58] on the effect of reputation on online auction

shows that sellers with low reputations typically get lower prices while sellers with higher reputations can collect more earnings. This is because buyers are willing to pay more for the perceived reliability and quality in services provided by sellers with high reputations. The expectation of reciprocity and retaliation provides the incentives and disincentives for a principal to exhibit trustworthy behavior.

The effectiveness of reputation systems relies on the quality of the feedback provided. Because of this, a problem with a reputation system is that very often, there is little incentive to provide feedback at the end of an interaction. The study in [58] finds that about half of the buyers on eBay provide feedback. In addition, fear of retaliation may discourage participants from providing negative feedback. In some cases, a participant may be blackmailed by another who threatens to retaliate with negative feedback. In [58], it is reported that on the average, about 1% of the feedback received by a seller is negative. It is suggested that perhaps, this low rate of negative feedback is due to the fear of retaliation. Another problem affecting the quality of feedback is the honesty of the feedback providers. For example, a group of sellers may work together to provide positive feedback to each other, resulting in all members of the group having good reputations.

Identity is another problem affecting the effectiveness of reputation systems [25]. Most reputation systems identify participants by their pseudonyms. However, participants with bad reputations can register a new pseudonym to start with a clean slate. As a result, newcomers may have to be distrusted, but on the other hand, if newcomers are distrusted, it will be difficult for them to establish good reputations. Some sites have specific measures to combat this problem. For example, eBay

requires a valid credit card in order to set up an account. However, these measures may not solve the problem completely. For example, a person with multiple credit cards can still set up multiple accounts. Moreover, the threats of identity thefts may render such measures ineffective.

3.2 Reputation Systems in Decentralized P2P networks

Decentralized P2P networks are widely used for sharing resources, such as file sharing. In fact, there are several well known P2P file sharing applications such as Napster [67], Gnutella [41], Freenet [43] and BitTorrent [18]. Given their popularity, P2P networks make an interesting study for trust establishment in decentralized environments since the results of this research can potentially create a huge impact. Moreover, there are many research efforts on using reputation to establish trust in P2P networks for resource sharing purposes [19, 40, 72]. This existing research serves as interesting case studies for building secure trust establishment methods. P2P resource sharing and online auctions share some common features that make reputation systems the popular choice. Both applications involve interactions among strangers. Besides that, the interaction patterns of both applications can be abstracted into a common model that can be described as follows:

1. A resource requester submits a query for a resource.

2. The requester receives a set of resource providers who can provide the resource.

3. A trust establishment step where the resource requester has to decide which provider to trust.

4. The requester obtains the resource from the provider.

Given the similarity in the interaction model, reputation systems can be used to help the resource requester determine which resource provider to trust.

While both applications share a similar interaction model, there is a major difference between them in terms of infrastructure. Most Internet applications have centralized administrations that provide the necessary services to support interaction among different participants. For example, the administration supports items 1 and 2 of the interaction model by maintaining a directory of resource providers, and also provides the communication linkage between the requester and the providers. The administration also assists in supporting trust establishment by providing reputation computation and storage services. In contrast, such centralized administration is absent from a decentralized network, with Napster being the rare exception. As a result, the participants of a P2P network have to cooperate to provide services that are similar to those found in applications with centralized administration. Supporting trust management is a big challenge, with problems involving the computation, storage and dissemination of reputation. In the absence of a centralized administration, how can the individual peers aggregate their feedback to construct the reputation of a particular peer? Where should reputation be stored, or more importantly, how can peers trust that the reputation storage has not been tampered with? How can one peer retrieve the reputation of another peer efficiently? Finally, how can one encourage peers to cooperate? Fortunately, there are numerous research efforts that are devoted to answering some of the challenges. This section reviews

some of these research results.

To address storage and retrieval of files, the P-Grid [5] provides a structure for organizing information in a P2P network. On top of that, [68] protects the storage by providing anonymity for the locations where files are stored. Since reputation is stored in files, these methods can easily be used to store and retrieve reputation.

In addition, research efforts [19, 40, 72] attempt to resolve the problem of computing reputation. In these schemes, each individual peer stores the feedback of each peer it has interacted with. This feedback can be combined efficiently from other peers in different ways.

In [19], the reputation of a provider is computed by a distributed polling algorithm. The file requester polls the other peers in the network for their opinion of a potential provider. Peers who receive the poll and who have previously interacted with the potential provider can respond by sending a vote, which contains either a positive or a negative response. The requester will make use of the votes to make a decision on whether to download a file from the potential provider. The outcome of the downloading will be used to rate the votes provided by the voters. This can be used to identify peers who provide quality votes and peers that do not. Having such information can help a peer decide which votes to give more weight to.

A prominent reputation system for P2P networks is the EigenTrust algorithm [40]. Like the previous scheme, each peer maintains a set of local reputations of peers it has previously interacted with. The EigenTrust algorithm is based on the PageRank [54] algorithm used by the Google search engine to rank web pages. This algorithm provides an efficient method to combine the local reputations stored in

each peer to obtain the global reputation of a peer.

In [72], the authors provide an interesting perspective of trust by using access control concepts. Each file is a protected resource in the P2P network. Each file is assigned two access thresholds, which represent two aspects of evaluating trust. The first aspect deals with how the host views the file requester (direct relationship) while the second views the performance of the requester in the P2P network (indirect relationship). To access a file, the requester has to satisfy these two threshold values. The direct relationship is built on previous interactions between the host and the requesting peer. This is computed based on how the host perceived the trustworthiness of the requester and the requester's contribution. The requester's contribution is computed based on the number of megabytes that had been transferred from the requester to the host peer. It captures the notion of whether the host owes the requester any downloads. The indirect relationship evaluates other peers' opinions on the requester based on how much they trust the requester and the amount of contribution made by the requester to these peers. The reputation storage problem is addressed by each peer storing the recommendations it received as well as the set of recommendations issued to other peers. A peer also maintains a blacklist of peers who it believes to have committed malicious acts. A peer will neither interact with peers in the blacklist nor recommend them. To access a file, the requesting peer is responsible for submitting the recommendations from other peers to the host, so that the host can compute the indirect relationship value. By taking contributions of peers into account, this scheme provides incentives for peers to share their files.

To encourage peers to cooperate in sharing files and providing reputation, incentives are often used. An example is seen in [72] where incentives are used to encourage peers to share their files. In [78], the computation of reputation includes a community context factor. This community context factor measures how active the participant is in providing feedback. Whenever a peer provides feedback, the peer's community context factor improves, which in turn improves the peer's reputation.

To deal with retaliation, [6] computes reputation based on the number of complaints. Suppose p provides bad service to q and r. Both q and r issue a complaint against p. To retaliate, p also issues complaints against q and r. p's reputation is defined as the number of complaints p has issued, multiplied by the number of complaints filed against p. In this case, the reputation of p is 4. In this scheme, a bigger reputation score represents less trustworthiness. A drawback of this scheme is that peers who filed genuine complaints will result in an increment of their reputations and hence make them less trustworthy. In this case, this is a disincentive for peers to file complaints. In [55], the scheme protects the identity of peers who provide feedback. By providing privacy, peers who provide negative feedback are protected against retaliation.

3.3 Security issues in P2P Reputation Systems

The use of reputation systems to establish trust in P2P networks causes a number of problems. A common problem is the lack of strong identity. In fact, a majority of the research efforts mentioned in the previous section use pseudonyms as

identities. The problem of using pseudonym has already been addressed earlier. An exception is the work of [19], which uses the hash of a public key as the identity of a peer. The use of public keys allows the peers to utilize authentication techniques to identify each other and to defend against authentication types of attacks, such as the man-in-the-middle attacks. However, this protocol is unable to protect against malicious peers with good reputations. In this situation, at least one peer will have to suffer from such attacker before the malicious act is discovered. When a malicious peer is identified by a victim, the victim can vote against the malicious peer in the future. Unfortunately, this research does not demonstrate the security properties of the scheme, such as how malicious peers with good reputations can affect the effectiveness of the reputation scheme.

The EigenTrust algorithm [40] addresses a variety of threat models, including the simple model of a malicious peer working alone, as well as the more sophisticated model of malicious collectives. A malicious collective comprises of malicious peers that always provide inauthentic files. Members of a collective will assign the maximum trust value to other members of the collective. Members of a malicious collective can even camouflage their intentions by providing authentic files in a certain fraction of cases when they are selected. The authors use simulations to demonstrate how well the EigenTrust algorithm can resist against these adversaries. The EigenTrust method is shown to be resilient against a malicious peer who works alone and has a certain degree of resilience against malicious collectives. The simulations show that malicious collectives have the maximum impact when 50% of the files provided are authentic. However, the effectiveness of the algorithm against

malicious collectives is due to the assumption of pre-trusted peers. The presence of pre-existing trust among some of the peers is a feature of the PageRank algorithm [54] which the EigenTrust algorithm is based on. It is unknown how EigenTrust will fare in the absence of such pre-existing trust. Moreover, in decentralized environments such as P2P networks, it is unreasonable to assume that peers with pre-existing trust are always present. Finally, the authors also simulate against the threat model of two malicious collectives cooperating with each other. Members of one group will always provide authentic files resulting in a good reputation. The malicious peers in this collective will then use their good reputation to boost the reputation of another collective. The simulation results show that this is a very effective way to attack the EigenTrust algorithm. Although a lot of attention is given to address various threat models, a criticism of the EigenTrust work is that these models seem to be derived in an ad hoc fashion. There is no explanation to show that these chosen threat models are adequate in addressing the security concerns.

While the work on EigenTrust attempts to simulate against a variety of threat models, there are some works that focus on a single aspect of security. For example, the schemes [6, 55] are only interested in combating retaliation, and [72] is only concerned with controlling access to resources. While these schemes do meet their respective security objectives, their threat models are simplistic and inadequate since there are a number of attacks that are not considered in the threat models of these works.

3.4 Research Challenges

The task of establishing trust in a decentralized network is difficult. Security adds to this challenge. Unfortunately, despite numerous efforts to address security, there is no common consensus on what security actually is. This is shown by the variety of different threat models in the research efforts, such as those mentioned in the previous section. The lack of a common definition of security also makes it difficult for one to determine whether a particular reputation system is secure or not. For example, while [6] is secure against retaliation, the authors have not demonstrated that this method is secure against other form of threats. Therefore, there is a need to define a common notion of security where all trust establishment methods can agree upon. This common definition can then be used to evaluate the security of any trust establishment methods. The security definition should preferably be independent of context, so that the definition can be applied not only on reputation systems, but other trust establishment methods as well.

Once the definition of security is agreed upon, the next challenge is to devise techniques to analyze the security of trust establishment methods. The primary objective of such analysis is to understand the security problems, with the hope that such understanding can lead to a more secure solution.

Chapter 4

Trust Establishment

The manner in which trust is established often depends on the context. For example, if an authority is present, trust management techniques [12, 11] can be used, whereas in the absence of an authority, the participating principals have to make trust decisions on their own, based on some information.

In general, the context can be described by either a closed world or open world model. In a closed world model, principals cannot participate freely. Instead, principals have to go through some registration process in order to participate. For example, in order to access a company's network resources, a person needs to be employed by that company. Each principal typically has a single digital identity that can be mapped to a set of roles, which in turn defines a set of responsibilities. Trust establishment decisions is typically dependent on the roles and responsibilities of a principal.

In contrast, in an open world model, any principal can participate at anytime. A common feature is the lack of a hierarchy of roles and authorities in this model. Because of this, trust establishment cannot be dependent on the roles of the principals. Although in theory, strong identities can be achieved through cryptographic means, such as public key cryptography, in practice, weak identities, such

as pseudonyms are used. Many Internet applications such as social networks sites or online auction sites, are examples of open world models. Trust management is not suitable for the open world model due to the lack of centralized authorities that dictate the policies for trust establishment. Therefore, many trust establishment methods are based on behavior patterns or existing relationships (in social networks).

This chapter presents an abstract trust establishment model that captures the important components of trust establishment, regardless of the context (open or closed world). Since the eventual goal is to be able to produce a common definition of security, the abstract trust establishment model can serve as a common basis for defining security.

4.1 Preliminary Trust Establishment Model

In Chapter 2, the establishment of trust is generalized into a two-step process: the computation of situational trust and decision making. The computation of situational trust depends on the application context. As mentioned in the review of existing research, situational trust can be computed using methods such as reputation, recommendation, credentials, utility, importance of the context, as well as the trust disposition of the decision maker. Depending on the application context, one or a combination of the above methods can be used for computing the situational trust. The computation of situational trust constitutes the semantics of trust establishment. The semantics of a trust establishment process provides the reasons

why trust can be established. For example, in a reputation system, the reason for trusting a principal is that the principal has behaved well in the past.

The decision making step of a trust establishment process involves evaluating the semantics of the trust establishment process. Using the reputation system as an example again, the decision making step evaluates whether the situational trust (reputation) of a principal is sufficient for trust to be established. For example, the evaluation may simply be a threshold function, such as the functions introduced in Marsh's work. In addition, the evaluation step may be used for selection purposes. For example, in a P2P file sharing application, it is possible that a resource requester receives multiple responses. The requester can compute the situational trust of all the peers who respond, and the evaluation process may select the peer with the best reputation.

Since the measurement of situational trust requires information, the gathering of information is an important part of the trust establishment process. Since information is gathered over a noisy environment, such as the Internet, it is important to consider the security of the information collected. The security of information is traditionally defined in security research as: confidentiality, integrity and authenticity. Confidentiality can be useful in protecting the contents of the information, while integrity ensures that the information is not modified either intentionally or unintentionally during transit. Authenticity identifies the source of the information and provides non-repudiable evidence of the information origin. In a closed world model, assuming the presence of authorities that uphold the rules for participation, these security requirements are usually sufficient. However, in an open world model,

non-repudiation becomes a challenge due to the lack of a centralized authority to uphold rules and regulations. Therefore, even if one has non-repudiable evidence that a principal makes certain claims, there is no authority to prosecute principals who do not fulfill their claims.

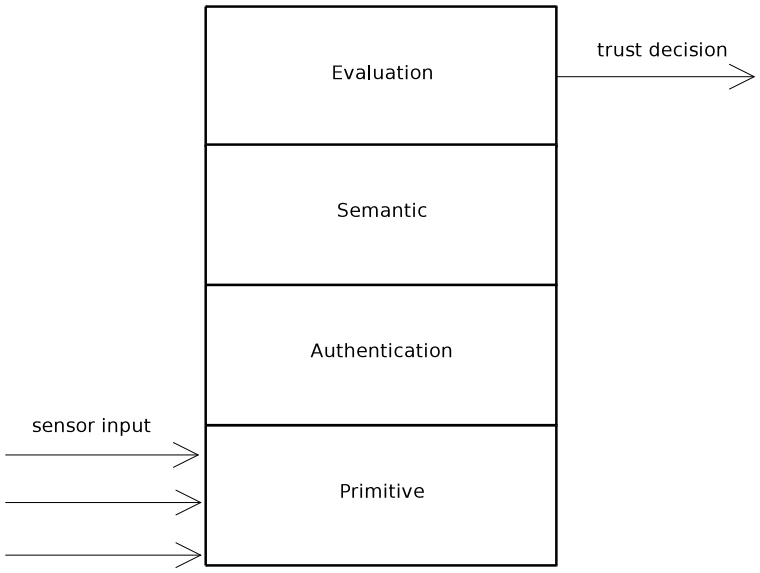

Figure 4.1: An preliminary trust establishment model.

Based on the above observations, a preliminary model can be described by Fig. 4.1. The trust establishment process can be thought of as a four layer model: an evaluation layer on top supported by a semantic layer below, followed by the authentication layer and finally, the primitive layer at the bottom.

At the top, the evaluation layer is simply a decision function, such as Marsh's cooperation threshold. It takes inputs from the semantic layer and compute a decision on whether to grant trust or not. It may also decide whom to trust.

The semantic layer is the reasoning engine behind trust establishment. The

semantic layer deals with the measurements of situational trust. As mentioned earlier, situational trust may be measured based on variety of factors, including risk, utility, importance and reputation. Situational trust may also be inferred based on trust propagation and aggregation rules.

The authentication layer provides some assurance on the integrity and authenticity of information. If confidentiality is a concern, it can be provided through encryption. In [39], Jøsang differentiates between trusting passionate entities with human-like capabilities and rational entities that are basically systems. Authentication is designed to verify the identity of passionate entities and to provide non-repudiable evidences about the information origin. In fact, many trust establishment methods target at passionate entities. However, trusting rational entities are also important because if a system is under the control of an adversary, the adversary can have access to secrets, such as private keys, that allow the adversary to masquerade the identity of an honest passionate entity. Therefore, the authentication layer may also verify the integrity of the computer system, in particular, the integrity of the application software. This can be done through remote attestation techniques such as [61, 66] and a hardware based monitor [56].

The primitive layer is a set of sensors that gather information from the environment. These sensors may be physical devices such as network interfaces that monitor network traffic, or they can be software processes that collect data over the Internet. Besides gathering information externally, information may also be obtained from local storage. Typically, local storage keeps information about past experiences.

Policies play an important role in trust establishment. In the trust establishment model, each layer is supported by a set of policies. The primitive layer can be supported by a set of policies that determine how to gather information, while the authentication layer is supported by another set of policies that determine the use of public keys. The policies at the semantic layer define situational trust, while the policies at the evaluation layer can be used as a decision function selector, such as, which decision function to use under which conditions.

The idea behind a layered-model approach is that each layer can be treated as an individual module, with the lower layer providing inputs to the upper layer. This form of modularity allows existing systems to inter-operate with each other. For example, one can customize the semantic layer to estimate situational trust using different methods (such as reputation), rather than the agent's disposition method as suggested by Marsh. The decision function (cooperation threshold model) at the evaluation model can remain unchanged as long as the new estimation method produces output within the range as required by the decision function. This is illustrated in Fig. 4.2. In Fig 4.2a, the semantic layer determines situational trust according to the trustor's disposition as proposed by Marsh. In Fig 4.2b, the decision function remains unchanged, but the semantic layer uses a reputation system to determine situational trust. This gives a trust establishment designer the flexibility to design solutions that can adapt to different contexts.

Another flexibility of the model comes from allowing composition of different methods at the semantic layer. This is especially useful if different systems can complement each other. For example, in [48], a recommender system is supplemented

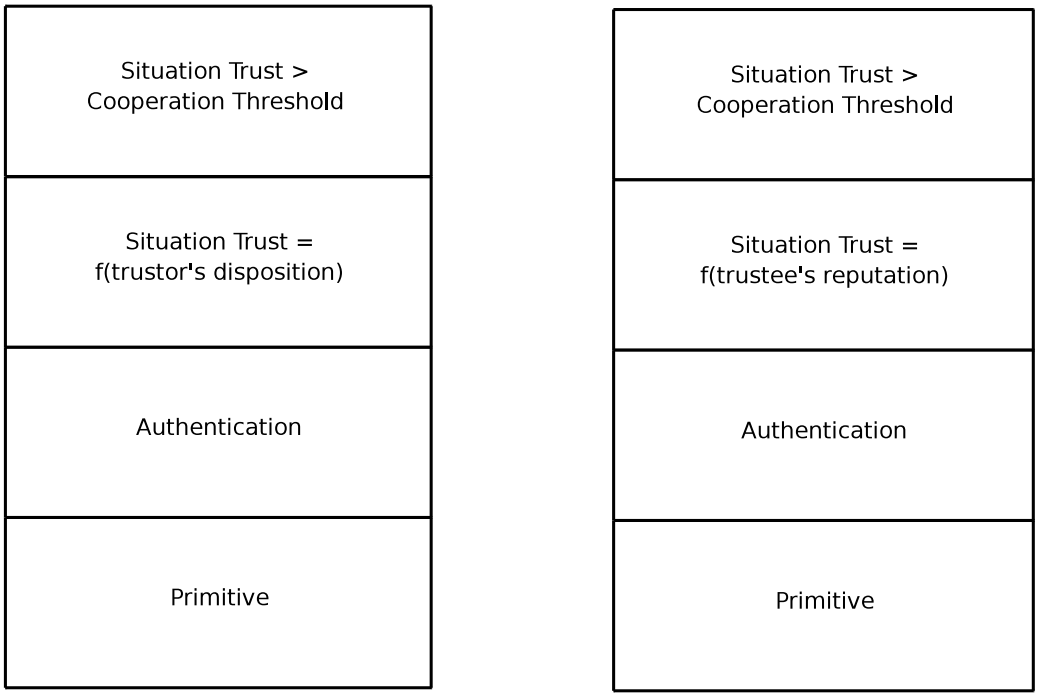

Figure 4.2: Example of how reputation systems can inter-operate with Marsh's model.

by a web of trust model to overcome the "cold start" problem of the recommender system. In this situation, one can think of the semantic layer as being partitioned as shown in Fig. 4.3. The aggregator is aware of the two methods below. It implements the method of [48] by selecting recommendations based on distances in the web of trust.

4.1.1 Semantic Layer

The semantic layer can be further refined as shown in Fig. 4.4.

The bottommost layer deals with information provenance. Authentication

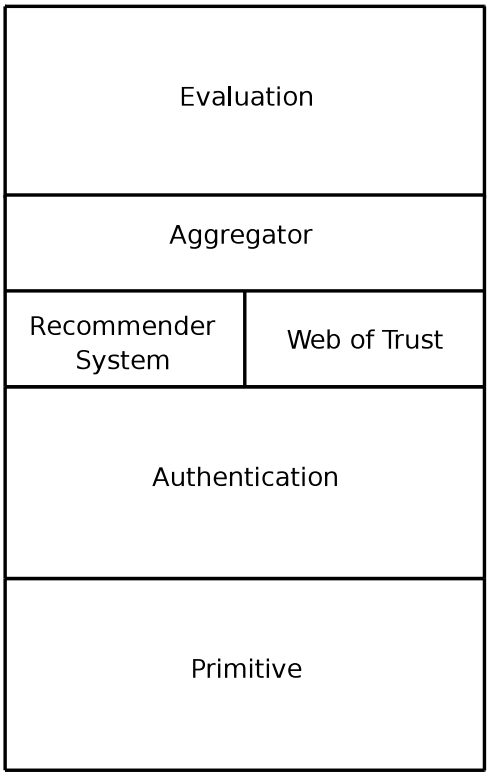

Figure 4.3: Example of composing the semantic layer with existing methods.

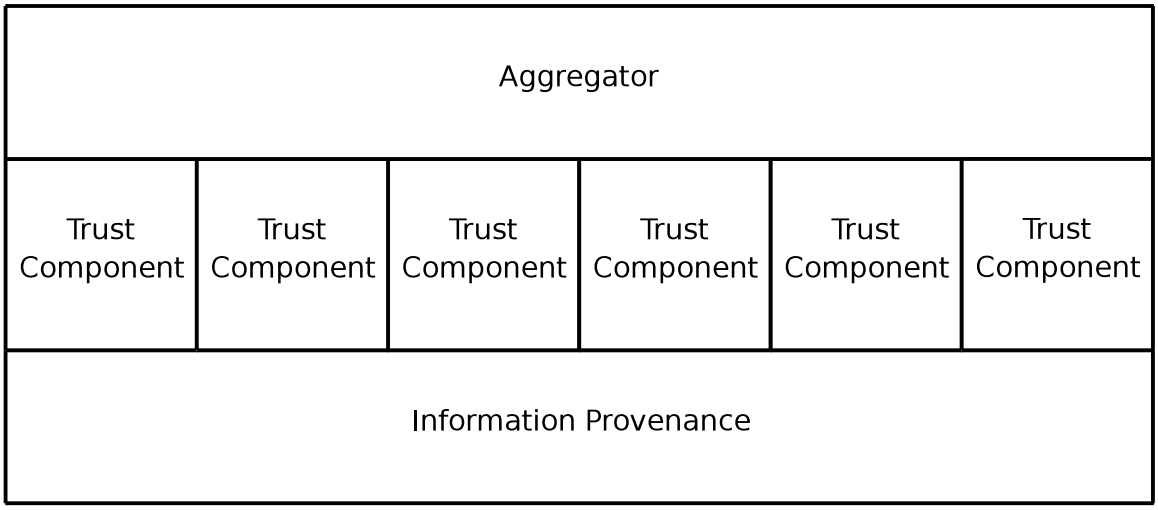

Figure 4.4: Further decomposition of the semantic layer.

only guarantees that information originates from a particular source. It does not provide any guarantee on the source's ability to provide accurate information. For example, if a principal x claims to provide a file f, the authentication layer can verify that x has indeed made the claim but the authentication layer cannot verify whether x can fulfill the claim of providing f. The aim of provenance is to establish some form of belief to such claims.

The subject of information provenance deals with the history of information. Of specific interest to trust establishment is the creator of the information and how the information has been modified since its creation. A way to establish the validity of information is via institutionalized trust. Information may be reliable if it originated from a trusted authority.

Above the provenance layer is another layer that comprises of one or more trust components. A component is a computational model for measuring a specific facet of the trust establishment process. For example, in the Marsh model, there is a component each for evaluating risk, competence, utility and importance. The trust establishment model accommodates multiple components. This allows different facets of trust to be combined.

Since these components occupy the same layer, they do not support each other directly. Instead, each component operates independently. This design allows different existing trust establishment methods to co-exist. For example, in Fig. 4.3, the trust components are a recommender system and a web of trust. The role of the aggregator layer is to take the outputs of each component and combine them in a meaningful way. For example, in Marsh's model, the aggregator simply gathers all

the measurements and passes them to the decision function at the evaluation layer, while in Fig. 4.3, the aggregator takes the recommendations from the recommender system and uses the trust values from the web of trust to compute recommendations as described in [48].

4.2 Expanded Trust Establishment Model

Based on the above discussions, the preliminary trust establishment can be expanded to the model shown in Fig. 4.5.

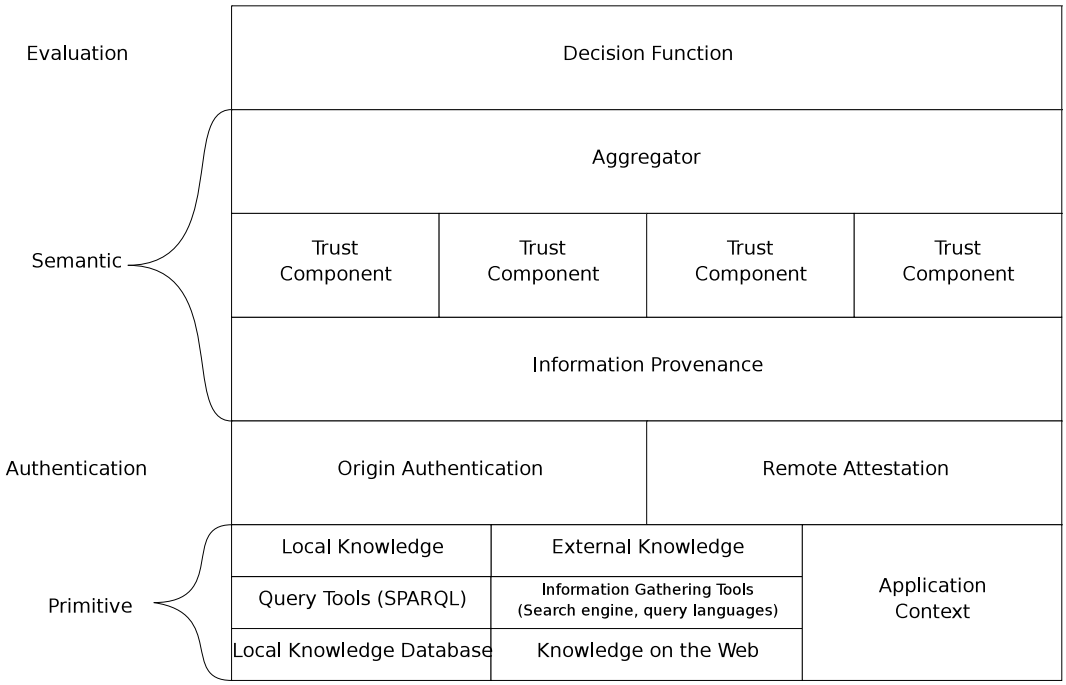

Figure 4.5: The expanded trust establishment model.

The evaluation layer at the top consists of a decision function, which remains unchanged from the preliminary model. The expansion of the semantic layer has already been described. The authentication layer is partitioned into two compo-

nents: origin authentication and remote attestation. Origin authentication verifies the information source and the integrity of the information, while remote attestation verifies the integrity of the software that provides the information.

At the bottom layer, the primitive layer is partitioned into three components. The first component deals with local knowledge. Local knowledge is stored in a local database, accessible through query tools, such as SQL. The second component deals with external knowledge that is on the web. This knowledge can be retrieved proactively using search tools. The external knowledge can also be gathered passively by collecting information submitted by other principals or through negotiating with other principals. The third component is the application context which guides the primitive layer through the information gathering process. Although the modularity of the model means that each layer hides the implementation details from other layers, the primitive layer needs to know the type of information to gather. For example, if the trust components in the semantic layer demands web of trust and recommender systems information, then the primitive layer can focus the retrieval efforts on information that are useful to the upper layers. Having some knowledge of the above layers also helps the primitive layer negotiate with another principal to agree on the type of information that is required, such as to use PGP PKI rather than X.509 PKI. The role of the application context is to provide a set of requirement specifications to guide the primitive layer through the information gathering process.

Each component of the model supports a null algorithm, which simply passes information through the component without any processing. For example, if the semantic layer consists of only one trust component, there is no need for an aggregator

to model this situation, the aggregator is said to implement the null algorithm.

One component that is missing in the model is the capacity for learning. Learning can help refine the components of the trust establishment model. Since the focus of this dissertation is on the security of trust establishment, learning is omitted. In the future, learning can be incorporated into the semantic layer.

4.3 Implementation Framework

The modular design of the trust establishment model provides a natural way to implement a trust establishment method. Each layer is implemented as a software component. A trust establishment implementation is simply a composition of different software components. A more challenging problem is to describe the interface of these software components so that different components can be "glued" together.

One approach is similar to programming with API, where each software component lists a set of inputs, outputs and their data types. However, this abstracts away the semantics of the information that flows in and out of each component. For example, consider the evaluation layer implementing Marsh's cooperation threshold function. The inputs will be a set of real numbers and the outputs will be a boolean value. The semantics of the inputs and outputs are lost. This can be a problem if the inputs or outputs involve complex data types. For example, to measure competence, the trust component may require some digital credentials issued by certain parties. It is difficult to describe the input in terms of data type in this case. The

type of the credential has to be specified, such as X.509 or SDSI/SPKI certificate. A data type for each possible type of credential has to be defined. To make things worse, it is also necessary to standardize the definitions of these complex data types, otherwise, it is difficult for different components to work together. For example, the developer of the evaluation layer may provide a definition of a digital credential data type while another developer of a trust component may provide another definition of the same digital credentials in a different manner. Although these two components define the same domain of knowledge (digital credentials), they cannot work together due to the difference in how the definitions are implemented.

Semantic web ontology provides a natural way to solve this problem. Different software components can be composed together by defining the interfaces using ontology. For example, the concept of digital credentials can be described using an ontology language, such as Web Ontology Language (OWL) [69]. In this way, each component can describe the inputs and outputs by referring to the ontology. To take it one step further, one can envision each layer to be a semantic web service. The OWL-S [71] language provides an ontology for describing services. A web service provides some computational functionality that can be utilized over the web. For example, a web service for the authentication layer may provide services to verify information integrity and origin. In order to invoke a service, the service provider will have to provide a set of invocation methods, and the service description for the service consumer. A semantic web service provides these descriptions using the OWL-S language so that the descriptions can be processed by a machine. Using OWL-S, a web service has to provide a service profile, service model and service

grounding.

The service profile describes the functionality of the service, the limitations, quality and conditions for using the service. This allows a service consumer to determine whether the service meets his needs.

The service model describes how to use the service. It specifies how to request the service and the outcomes that will occur under specific conditions. This set of descriptions allows a service user to have a more detailed information about the service, to compose multiple services to perform a task, to coordinate different events, and to monitor the execution of the service.

While the service model describes how to use the service, the service grounding provides a concrete description for using a service. It specifies the communication protocol, message formats and the addressing issues.

With each layer being implemented as a service, the trust establishment involves identifying the suitable services for each layer and composes these services to form a trust establishment method.

The semantic web also provides a reasoning platform for supporting provenance. For example, the TRELLIS system [27] provides tools to allow collaborative annotation of information. Users can add annotations to web resources which may indicate the accuracy of the resource itself or of the creator of the resource. These annotations can allow other users to derive how much to trust the information contained in a web resource. Besides annotation, the semantic web also supports the derivation of trust from social networks [28]. Trust derived from social networks can be used to determine the trustworthiness in the creator of a web resource.

4.3.1 Example

To illustrate the implementation framework, consider the scenario where Alice wishes to search for an interior designer. To do so, she uses a web service that advertises her request on the Internet, as well as her list of requirements, among which may include the credentials of the designer and references.

Suppose Bob and Carol respond to the advertisement with their signed credentials and signed references as required by Alice. The credentials and references can be expressed as statements using OWL ontology. For example, the trust ontology [1] can be used to express the reference information, such as, "Joe trusts Bob highly regarding interior design." The statements can then be represented as named graphs [15], which can be signed. These pieces of information are passed to the authentication layer.

The authentication verifies the source and integrity of the statements. For example, if Bob claims the statement "Bob has $Bob_{credential}$", then the authentication layer verifies that the statement originates from Bob and the content of the statement has not been changed. Depending on the type of key used to sign the statement, the relevant authentication service is selected. For example, if Bob's statement is signed using RSA key and Carol uses DSA signatures, then the web service that is capable of verifying RSA signature is chosen for Bob's statements while a service capable of verifying DSA signature is chosen for Carol. Note that the authentication in this example does not deal with timeliness of information. An adversary may be

[1] Trust ontology: http://trust.mindswap.org/ont/trust.owl

able to replay old statements from Bob. To provide a freshness guarantee, Alice will have to instruct the responders to send the relevant claims to a trusted web service that authenticates the timeliness of the information using more sophisticated authentication protocols, such as Kerberos [52].

Upon successful authentication, the authenticated statements are passed to the provenance layer. The provenance layer can verify the credential statements, by retrieving the credentials through the URI in the statements (recall that the credentials are expressed in OWL). The credentials can be examined to verify their validity. For example, a web service specializing in interior design schools can check whether the issuer of Bob's credentials is an accredited institution and whether the credentials are still valid. The provenance web service may not be able to verify the validity of the reference information. Consider the statement "Joe trusts Bob highly regarding interior design." While an authentication service can verify that Joe has indeed claimed that statement, a provenance service is unable to verify whether Joe is telling the truth. Information provenance requires trust anchors that determine which principals can be trusted. Therefore unless there is information available that establish Joe as an authority on interior design (perhaps Joe is a known expert), or some trusted principal can vouch for Joe, nothing can be known about the truth of the references. For example, if the social network of Alice is known, and Joe is a friend (neighbor) of Alice in the social network, then the provenance layer may infer that Joe is telling the truth, especially if Joe is a friend of Alice and is unlikely to lie to Alice.

At the trust component layer, suppose Alice chooses a reputation system and a

recommender system component. The reputation system provides the reputations of the credential issuers, while the recommender system provides similarity measures between Alice and the reference providers. Suppose the reputation system rates Bob's credential issuer with 0.8 and Carol's credential issuer as 0.6. Also suppose the similarity measure between Alice and Joe, who provides reference for Bob, is 0.7, where the bigger the rating, the more the similarity. The similarity measure between Alice and James, who provides reference for Carol, is 0.9. The aggregator combines these ratings to measure Bob and Carol's competence by multiplying their respective ratings from the two trust components. This results in Bob having a competence rating of 0.56 while Carol's competence rating is 0.54. The evaluation layer selects Bob due to Bob having a higher competence.

Chapter 5

Security of Trust Establishment

5.1 Robust Trust Establishment

The bottom layer of the trust establishment model represents the environment where information is gathered. The gathered information may contain noise since anything can happen to information as the information moves across the network. Internal knowledge too can be corrupted if an attacker can compromise the local storage and modify the information. For simplicity, lets assume that local storage is secure, using secure storage solutions such as a trusted platform [2] or an encrypted storage [10, 53]. The authentication layer attempts to remove some of the noise from the external information by removing security problems, such as information tampering, information replay and man-in-the-middle attacks. Also, the use of a PKI can also provide confidentiality services and strong identities. In addition, remote attestation can also verify the integrity of the software running on machines that participate in the networks. When attempting to define security, the focus will be on the semantic layer and above. In practice, authentication services are not always available. However, security research on authentication is well established and the security problems of authentication are well known [14, 51, 30, 70] and

well researched. As such, authentication challenges are not the main concern of the dissertation. Instead, the authentication layer is assumed to employ one of the well known techniques to address authentication challenges.

At the semantic and evaluation layers, the security challenge is to prevent adversaries from gaining trust. Trust components rely on information to make trust decisions. As explained in the previous chapters, even in the presence of an authentication layer, adversaries may still be able to manipulate information so that he can be perceived as trustworthy. Therefore, security of trust establishment can be based on how easy it is for adversaries to gain trust.

In an ideal setting, a trust establishment method should establish trust with an honest principal and it should distrust a malicious principal. In reality, there are false positives (malicious principals identified as honest) and false negatives (honest principals identified as malicious). The presence of false positives and false negatives is a security concern since it allows malicious principals to be trusted and it denies honest principals from being trusted. Therefore, one can think of defining security in terms of how well a trust establishment process can distinguish between honest and malicious principals. The lower the false positive and false negative rates, the more secure a trust establishment method is.

More formally, given a context C, let $\Pi(p, q)$ be a trust establishment method with p, q as principals, such that one of them is honest while the other is malicious. The ultimate goal of Π is to determine which principals can be trusted. The context C provides information that supports Π's decision making. Formally, a generic trust establishment method Π can be described as follows:

1. Randomly select two principals p and q, from C, such that one of them is honest while the other is malicious.

2. Given p, q as inputs, $\Pi(p, q)$ repeatedly make any number of queries to C.

3. $\Pi(p, q)$ outputs the identity of the malicious principal.

In [1], the term, "robustness" is defined as follows:

Definition 2 *Robustness: the degree to which a system operates correctly in the presence of exceptional inputs or stress environmental conditions.*

In the context of trust establishment, the robustness of a trust establishment method, Π, is the degree to which it establishes trust correctly in the presence of malicious principals. This degree of correctness corresponds to the probability that Π outputs the correct answer. Π is said to be robust if it can output the correct answer with a high probability. Conversely, Π is not robust if the probability is low. Since the upper bound of the probability is 1, the closer the probability is to 1, the more robust Π is. Similarly, the probability of 0 can be used as the lower bound. However, one can achieve a tighter bound by considering the case of a random trust establishment method which guesses the output randomly, based on uniform probability distribution over random coin tosses. In this case, the random trust establishment method tosses a coin randomly. If the outcome is heads, the method outputs p, otherwise, it outputs q. The possible outcomes are $(Head, p \textit{ is malicious})$, $(Tail, p \textit{ is malicious})$, $(Head, q \textit{ is malicious})$, and $(Tail, q \textit{ is malicious})$. Therefore, the random trust establishment method produces the correct output with the probability of $\frac{1}{2}$. Any other trust establishment method must be more robust than the

random method, otherwise, it defeats the computational efforts involved in establishing trust. Therefore, the probability of $\frac{1}{2}$ is used as a lower bound in the definition of robustness.

Although the above definition only involves two principals p and q, the definition can be extended to include multiple principals. In the general case, consider a set of principals where n are honest and m are malicious. Then the $\frac{n}{n+m}$ is the lower bound probability for measuring robustness.

The goal of an adversary is to lower $\Pi's$ probability of producing the correct output. The more robust Π is, the more resilient it is. Therefore, security of a trust establishment method can be defined by the robustness of the method.

To understand the robustness of a trust establishment method, it is necessary to understand the capabilities of the adversaries. The application context in which a trust establishment method is used defines a set of actions that participants can perform. Since the adversaries are participants, they are capable of performing these actions. It is interesting to know how adversaries can take advantage of these actions to gain trust. Therefore, given Π and a set of actions permitted by an application context, the interesting question is whether there exists any adversarial algorithms, where each algorithm is a sequence of actions permitted by the context, such that Π is not robust. Besides the set of actions permitted by the application context, the adversaries are also capable of actions that are outside the scope of the context, such as compromising a participant's computer system or performing denial of service attacks. However, such attacks fall under the domain of systems and network security where there may be appropriate solutions to deal with such

attacks. For example, research efforts in trusted computing and secure programming may help to protect systems from being compromised. On the other hand, there are very little research efforts on how adversaries can take advantage of the permitted actions to gain trust. In fact, as mentioned in the previous chapter, most research efforts treat security of trust establishment in an ad hoc manner by only addressing specific problems without much justification as to why such problems are chosen and whether their works are secure against other types of attacks. As such, the analysis of the robustness of a trust establishment method focuses only on how adversaries can exploit the allowed actions to gain trust.

5.2 Security of Reputation Systems

As mentioned previously, reputation systems are popularly used for establishing trust in decentralized environments. Therefore, reputation systems provide an interesting subject for security analysis, as well as for illustrating the analytical techniques.

Based on the above definition of security, it is easy to show that reputation systems are not robust. Consider the reputation system, $\Pi_{rep}(p, q)$ for some context C. The generic definition of a reputation system can be described as follows:

1. Query C about p and q's reputation.

2. If the reputation of p is greater than the reputation of q, output p. Else, output q.

To define an adversary algorithm, the first step is to model the application con-

text. Since the interest is in decentralized environments, P2P file sharing is chosen as an example for illustrative purposes. The context can be represented by a set of principals P. Each principal $p \in P$ maintains a state variable $outcome_p = \{+, -\}$. This state variable represents the outcome of p's participation in an action. If the outcome of performing an action is positive to p, then $outcome_p = +$. Similarly, if the outcome is negative to p, then $outcome_p = -$. The model can be completed by defining the actions permitted by the application. Without going into specific details pertaining to a particular P2P application, P2P file sharing applications generally are comprised of the following basic actions: $Join(p)$, $Leave(p)$, $+\_Action(p, q)$ and $-\_Action(p, q)$.

The $Join(p)$ and $Leave(p)$ actions describe a principal, p, joining and leaving the C. $+\_Action(p, q)$ and $-\_Action(p, q)$ describe principal p providing a positive and negative action towards principal q respectively. A positive action is one that demonstrates good behavior as intended by the application, while a negative action is one that shows malicious intent. As an example, consider a simple file sharing scenario where p requests to download a file from q. This request action can be interpreted as a positive action (since there is nothing malicious about requesting a file download from the point of view of a file sharing application), represented by $+\_Action(p, q)$. The response from q can be either $+\_Action(q, p)$ if q sends an correct file, or $-\_Action(q, p)$ if q sends an incorrect file.

Each action is associated with a set of pre- and post-conditions. The pre-conditions of an action specify the conditions in which an action can be carried out. These conditions are defined in terms of the states of the application. The

post-conditions of an action described the new state of the system as a result of performing the action. An action, together with its pre and post conditions can be expressed in the form of Hoare triples [34]: $\{X\}Y\{Z\}$, where Y represents an action, X and Z represent the pre and post conditions of the action respectively. Formally, the Hoare triples of the actions in application C can be described as:

$$\{p \notin P\}Join(p)\{p \in P\} \tag{5.1}$$

$$\{p \in P\}Leave(p)\{p \notin P\} \tag{5.2}$$

$$\{p, q \in P\} + \_Action(p, q)\{outcome_p = +, outcome_q = +\} \tag{5.3}$$

$$\{p, q \in P\} - \_Action(p, q)\{outcome_p = +, outcome_q = -\} \tag{5.4}$$

After defining the application context, similar steps can be used to define the trust establishment method within the context. This example describes a reputation system, Π_{rep}. It consists of a set R where $r_p \in R$ represents the reputation of a principal $p \in P$. The actions of a generic reputation system can be described by the following: $GetRep(p)$ and $Update(p, q)$. The action $GetRep(p)$ returns the reputation of a principal p, while $Update(p, q)$ represents the action where p updates the reputation of q to obtain a new reputation r'_q. Formally, these actions can be described using Hoare triples:

$$\{\}GetRep(p)\{\} \tag{5.5}$$

$$\{+\_Action(q, p) \vee -\_Action(q, p)\}Update(p, q)\{(r'_q < r_q) \vee (r'_q = r_q) \vee (r'_q > r_q)\} \tag{5.6}$$

The $GetRep(p)$ action does not affect any state of the reputation system. The $Update(p, q)$ action requires q to have performed either a positive or negative action. The result of an update is such that the new reputation of q, r'_q, may increase or decrease, or even remain unchanged to capture the situation where p does not provide feedback. Notice that $\{-\_Action(q, p)\}Update(p, q)\{r'_q > r_q\}$ and $\{+\_Action(q, p)\}Update(p, q)\{r'_q < r_q\}$ are valid actions. This describes the problem where users may not provide feedback honestly.

Having defined both the application and the trust establishment method to be used within the application, the next step is to define the coupling of actions from the application and trust establishment method. A coupling is a rule, represented by $a_i \rightarrow a_j$. The rule states that if action a_i occurs, then eventually, a_j will occur. The rule does not imply that a_j will occur immediately after the completion of a_i. It only states a_j will occur some time after the completion of a_i. In the example, the following rule is defined:

$$(+\_Action(p, q) \vee -\_Action(p, q)) \rightarrow Update(q, p) \tag{5.7}$$

This rule states that if p executes a positive or negative action to q, eventually q will update p's reputation. This rule is consistent with the pre-conditions of the $Update(p, q)$ action. All actions, positive or negative, lead to an update of reputation. The event where q does not provide feedback is modeled by an $Update(p, q)$ action resulting in $r'_q = r_q$.

Based the above models, an adversary, $adv \in P$ can perform the following algorithm $ADV()$ for the application context C and Π_{rep}:

```
ADV(){
        Join(adv);
        for all p in P:
                GetRep(p);
        end for;

        Repeat
                +_Action(adv,q), for any principal q;
        Until GoodEnough;

        -_Action(adv,q'), for any principal q';
        Leave(adv);
}
```

GoodEnough is a condition chosen by *adv* to determine when to stop performing positive actions. The condition is satisfied if Π_{rep} cannot distinguish between *adv* and an honest principal. The execution of a positive action will lead to the update of *adv*'s reputation as defined by rule 5.2. If the majority positive actions lead to increase in reputation, repeated execution of positive actions will lead to an increase in *adv*'s reputation in the long run. Since reputation information is public (obtained by performing *Getrep*()), an adversary knows exactly how much reputation he is required to accumulate in order to be indistinguishable from an honest principal. Therefore, an adversary knows when *GoodEnough* is satisfied.

When the adversary exits from the loop, the adversary has already accumulated sufficient reputation such that Π_{rep} cannot distinguish between an adversary and an honest principal. Therefore, given *adv* and *hon*, where *hon* is an honest principal, $\Pi_{rep}(adv, hon)$ is unable to output the correct identity of the adversary with a high probability.

5.3 Application Specific Exploits

Recall that one of the security concerns is the accuracy of the information used for making trust decisions. The analysis in the previous section shows that adversaries can control their reputation scores to gain trust. The analysis is done without details of the application context. Instead, the actions of the application are simply described as positive or negative actions. Therefore, the adversary algorithm, $ADV()$ from the previous section is due to the flaw of reputation systems in general. $ADV()$ exists whenever reputation systems are used, regardless of the application context.

This section builds on the previous section by considering application context in greater details. By applying the same techniques, flaws in application contexts can be discovered. Using P2P file sharing as an example, the model for application context C from the previous section can be expanded by replacing $+\_Action(p, q)$ and $-\_Action(p, q)$ with concrete protocol actions.

Let p be the file requesting peer and q be the file providing peer. Also, let s represents the system. The system is an abstraction of the essential services available

in a P2P file sharing application. For example, different P2P file sharing applications have different protocols for finding, storing and downloading files. In this model, these services are assumed to be provided by the system, s. The definition of s does not place any assumption about the structure of the P2P network. The network may have a semi-decentralized structure, such as the Napster [67] network architecture, where there is a centralized server that provides file location services. In the case of Napster, s represents the centralized server. The network may also be a totally decentralized network, such as Gnutella [41] which requires the cooperation of peers to locate files. In this situation, s is the group of peers that cooperate to provide the services ($s \subseteq P - \{p, q\}$).

Assuming that peers p, q, s have already joined the P2P network ($p, q, s \in P$). Then a generic P2P file sharing application can be described as a series of interactions between p, q, s in the sequence shown in Fig 5.1.

$$
\begin{array}{rrl}
+1. & p \rightarrow s: & Query \\
+/-2. & s \rightarrow p: & Response \\
+/-3. & p: & q = \Pi_{rep}(Response) \\
+4. & p \rightarrow q: & Request \\
+5. & q \rightarrow p: & Upload \\
+/-6. & p: & Verify \\
+7. & p \rightarrow s: & Update(p, q)
\end{array}
$$

Figure 5.1: A generic P2P file sharing protocol.

Each of the steps in Fig. 5.1 is an action of the P2P file sharing application.

Peer p sends a query to the system for a file. The exact details of the file does not matter to the security analysis. The system responds with a set of peers who have the file. In step 3, p engages the reputation trust establishment method to identify the peer to download the file from. Steps 4 and 5 correspond to the file transfer phase where p requests to download from the provider q and q uploads the file to p. Let "NULL" represent a special file that represents an event where q does not upload anything to p. In this sense, an upload is always guaranteed to take place. In step 6, p verifies the downloaded file from q and updates q's reputation in step 7. The verification in step 6 checks that the content of the downloaded file is correct. A file is correct if its content is what p expects. Otherwise it is incorrect. For example, if p downloads a song, then p expects to hear the song in step 6. The terms "authentic" and "inauthentic" are also used in [40] to describe a file that is correct and incorrect respectively. However, to avoid confusion with authentication services at the authentication layer, these terms are not used in this dissertation.

The sequence of actions in Fig. 5.1 constitutes a single transaction. Within a single transaction, the actions of the transaction take place in sequence, that is:

$$\text{Step } i \rightarrow \text{Step } i+1. \tag{5.8}$$

No peers in P is able to execute these actions out of sequence. However, recall that step $i+1$ does not necessary take place immediately after the completion of step i. In fact, there may be a time lag in between two consecutive steps of the protocol. In addition, a peer may be involved in multiple transactions such that the executions of the actions from different transactions may be interleaved by a peer. Timing and

concurrency issues are often red flags for race conditions. Therefore, the analysis will have to check for race conditions. To facilitate the checking for race conditions, define t_i to be the time just before step i of a transaction is about to be executed and t'_i be the time just after the execution of step i. Then, the execution of a transaction must satisfy the following timing constraint:

$$t'_i \leq t_{i+1}. \tag{5.9}$$

A generic protocol is sufficient for security because the implementation details of each individual protocol do not affect the security analysis. For example, the results of the security analysis is independent of how files are distributed or how files are located. It is assumed that s provides these services. However, it is **not** assumed that s provides these services reliably. The advantage of using a generic protocol instead of specific application protocols is to allow the analytical results to be applicable to as many specific applications as possible. In this way, the analysis results can be applied to any P2P file sharing applications that follow the same pattern described by the generic protocol. In fact, the results are also applicable to any applications that can be abstracted into the above pattern. For example, the protocol for resource sharing in P2P networks can be described by the same generic protocol. The only difference is in step 5 where a requested resource is granted instead of $Upload()$.

In Fig. 5.1, each step is annotated to indicate whether it is a positive or negative action. The annotations are done from the perspective of p. Therefore, all

actions originating from p are positive actions. The rest of the actions can either be positive or negative. Step 5 is considered a positive action because the upload is always guaranteed to take place. The outcomes of steps 3 and 6 depend on the output of these steps. p expects q to be an honest user as selected by the reputation system in step 3. The outcome for step 3 is positive if q is indeed honest and the outcome is negative if q is an adversary. Similarly, a successful verification in step 6 results in step 6 being a positive action and a failed verification means that step 6 is a negative action.

The goal of any rational participant of an application is assumed to achieve positive outcomes, otherwise, it makes no sense to participate. Therefore, the security goal of an application can be described informally by the invariant: "p believes that the outcome of each action is positive". To ensure that the invariant holds, p must be able to establish that an action leads to positive outcome. The goal of the analysis is to verify that this invariant holds throughout the execution of a transaction. Before expressing this invariant formally, it is necessary to introduce some extra notations and to explain the approach to analysis.

The establishment of beliefs is commonly used to verify security protocols. A main usage of belief logic in security is to verify authentication protocols. Among the various methods, BAN logic [14] is perhaps the most well-known method. BAN logic expresses authentication goals as a set of beliefs, such as "p believes that q says x" expresses the belief in the origin of an item x. However, the language of BAN logic only supports the analysis of authentication protocol. To support reasoning about the outcomes of an action, new notations and inference rules have to be

introduced.

5.3.1 Brief Review of BAN

As the name implies, Extended BAN (ExBAN) is an extension of BAN. Therefore, before introducing ExBAN, it is necessary to understand some of the fundamental concepts of BAN. First, the objective of BAN is to prove whether a particular belief can be established. For this purpose, the "believes" notation, $\models$, expresses the establishment of a belief. More specifically:

Definition 3 $p \models x$*: p believes x, or p would be entitled to believe x.*

This notation means that the principal p may act as though x is true, where x can be anything, including predicates, information and digital artifacts. The concept of belief is time sensitive since belief can change over time. In BAN logic, any established beliefs are only valid for the current transaction in which they are established.

Authentication typically involves exchange of challenges and responses over the network. When a principal receives a message over a network, this can be expressed using the "sees" notation:

Definition 4 $p \triangleleft x$*: Someone has sent a message containing x to p, who can read an repeat x.*

If the origin of a message can be identified, this can be expressed by the "once said" notation:

Definition 5 $p \mid\sim x$*: p at some time sent a message containing x. It is not known*

whether the message is sent some time ago or during the current run of the protocol, but it is known that p believed in x at the time p sent the message.

The important point about the "once said" notation is the time when the message is sent is unknown. If the x is not sent during the current run, then there is a possibility that p may no longer believe in x during the current run. To express messages that are valid in the current run, the "fresh" notation is used:

Definition 6 $\sharp(x)$*: x has not been sent in any messages at any time before the current run of the protocol.*

Another relevant notation is the jurisdiction notation, $\models\!\Rightarrow$. The jurisdiction notation in BAN logic expresses the powers of an authority:

Definition 7 $p \models\!\Rightarrow x$*: p has jurisdiction over x.*

This can be interpreted to mean that the principal p is an authority on x and should be trusted on this matter. The concept of jurisdiction is similar to institutionalized trust mentioned in Chapter 2.

Jurisdiction rule allows beliefs to be established:

Definition 8 *Jurisdiction rule: If p believes q to be an authority over x* ($p \models (q \models\!\Rightarrow x)$)*, and p is convinced that q believes in x, then p also believes in x:*

$$\frac{p \models (q \models\!\Rightarrow x), \quad p \models (q \models x)}{p \models x}.$$

The jurisdiction rule expresses the type of trust that is similar to Barber's view of trust as a form of expectation. Principal p is willing to believe in x because

p expects q to dispense his duties faithfully. Since the purpose of BAN is to serve as a tool for analyzing authentication protocols, the jurisdiction notation is typically used to declare the identity of certificate authorities, while the jurisdiction rule is typically used to establish beliefs in public key ownership:

$$\frac{p \models (ca \Rightarrow \overset{pk_q}{\mapsto} q), \quad p \models (ca \models \overset{pk_q}{\mapsto} q)}{p \models \overset{pk_q}{\mapsto} q}$$

This means that if p believes ca to be the authority over public key ownership, and if p knows that ca believes that q owns pk_q ($\overset{pk_q}{\mapsto} q$), then p believes that q owns pk_q. $p \models (ca \models \overset{pk_q}{\mapsto} q)$ can be established when p verifies q's public key certificate issued and signed by ca.

When a principal p receives a message from q, the authentication layer can verify the message origin and integrity, as well as timeliness of the message based on cryptographic keys and a random nonce. This is represented in BAN by the message meaning rules and the nonce verification rule:

Definition 9 *Message meaning rule (using public key cryptography):*

$$\frac{p \models \overset{pk_q}{\mapsto} q, \quad p \triangleleft \{x\}_{pk_q^{-1}}}{p \models q \mid\sim x}.$$

Definition 10 *Nonce verification rule:*

$$\frac{p \models \sharp(x), \quad p \models q \mid\sim x}{p \models q \models x}.$$

The message meaning rule explains how to derive the belief in the origin of messages. In Definition 9, if p believes that pk_q is the public key of q ($\overset{pk_q}{\mapsto} q$) and p

sees a message signed with the signature key of q, then p believes that q once said x.

While the message meaning rule captures the idea that the successful verification of a signed message leads to the conclusion about the message origin and integrity of the message, it is unknown whether the message is current or is it a replayed message. To derive the belief in the timeliness of the message, the nonce verification rule states that if p believes that the message x is fresh, and p believes that q once said x, then p believes that q believes in x. p can determine that x is fresh if x contains a random nonce that is known to p to be recent.

5.3.2 ExBAN (Extended BAN)

While BAN is designed for analyzing authentication protocols, the idea of ExBAN is to use similar techniques to reason about trust establishment within an application context. To do so, ExBAN introduces new notation and rules for analyzing trust establishment. Referring to the trust establishment model, authentication is a component of the model, whose role is to verify the origin, integrity and timeliness of information. Therefore, BAN is relevant to trust establishment as a tool for analyzing the authentication layer. By using BAN, one can determine the authenticity of the information that reaches the semantic layer. The role of ExBAN is to continue the analysis at the semantic layer.

To represent trust in a principal, ExBAN introduces the concepts of trusted predicate and trusted principal:

Definition 11 *$T(q, x)$: q is trusted on x.*

Definition 12 *$p \models T(q, x)$: p trusts q over x.*

Jurisdiction in BAN expresses a special form of trusted principal. As explained earlier, jurisdiction is simply an expression of institutionalized trust in an authority (an authority over key and name binding in BAN). Therefore, if p believes that q has jurisdiction over x, this implies that p trusts q on x:

$$p \models (q \Rightarrow x) \Rightarrow p \models T(q, x). \tag{5.10}$$

However, the converse is not true. That is:

$$p \models T(q, x) \not\Rightarrow p \models (q \Rightarrow x). \tag{5.11}$$

This is because p may decide to trust q on x, even though q is not an authority over x. For example, consider a social network scenario where p trusts q because q is a friend. In decentralized environments, trusted principals is more important than jurisdiction due to the lack of authorities in these environments. The presence of trusted principal is important for trust establishment, otherwise, the trust establishment process may involve traversing a potentially endless chain of trust relationships. For example, in a reputation system, to determine which principal to trust depends on the robustness of the reputation system. The robustness of reputation systems in turn relies upon the trustworthiness of the principals who provided feedback. One can then continue to determine the trustworthiness of each principal who provided feedback and so on. However, establishing the trustworthiness of each principal may not be possible since there may not be sufficient information to verify

each and every principal that provides feedback. Moreover, this is a potentially computationally intensive process. To avoid this work, most trust establishment methods terminate at a particular stage and take a "leap of faith." For example, many reputation systems do not go beyond verifying each of the principals who provided feedback. Instead, it is taken as faith that the majority of the feedback is accurate such that the inaccurate feedback does not have significant impact on the actual reputation. In this example, the principals who provide feedback are the trusted principals. Obviously, such "leaps of faith" can be exploited by adversaries.

In authentication protocols, the honesty of a principal is not a problem. A digital signature serves as a piece of non-repudiable evidence about the origin and integrity of the information. More importantly, authentication protocols assumed that there are authorities that can prosecute an errant principal based on the non-repudiable evidences. In the absence of an authority in decentralized environments, non-repudiable evidence is not useful due to the lack of authorities to enforce good behavior. Therefore, in decentralized environments, honesty of principals becomes a concern. To model the honesty aspect, a stricter definition of belief is required. Let K_p represents the set of knowledge belonging to p. Then honesty is modeled by:

Definition 13 *$p \overset{h}{\models} x$: p is honest about x, such that if $p \overset{h}{\models} x \Leftrightarrow x \in K_p$.*

To establish the honesty of q's belief, ExBAN introduces the trust inference rule:

Definition 14 *Trust inference rule:*

$$\frac{p \models T(q,x), \quad p \models q \models x}{p \models q \overset{h}{\models} x}$$

The trust inference rule explains that if p trusts q on x and p believes that x originates from q and x is current, then p believes that q is honest about q. From Definition 13, p also believes that x belongs to q's knowledge.

5.3.3 Application Security Analysis Using ExBAN

Returning to the problem of analyzing the robustness of reputation systems in P2P file sharing, the invariant introduced earlier can be expressed using ExBAN:

For $1 \leq i \leq 7$:

$$p \models (outcome_p = +) \text{ at } t_i \Rightarrow p \models (outcome_p = +) \text{ at } t'_i$$

$$\wedge \tag{5.12}$$

for $1 \leq i \leq 6$:

$$p \models (outcome_p = +) \text{ at } t'_i \Rightarrow p \models (outcome_p = +) \text{ at } t_{i+1}.$$

The first part of Invariant 5.12 states that if p believes that the outcome is '+' before executing step i, then p still believes that the outcome is '+' after executing step i. The second part states that if p believes that the outcome is '+' after executing step i, then p still believes that the outcome is '+' just before executing step $i + 1$.

In Fig. 5.1, the steps of a P2P file sharing application are annotated to indicate whether the step results in a positive or negative outcome. The annotations are created from the perspective of p. Therefore, steps originating from p result in positive outcome. For these steps, the outcome is '+' before and after executing these steps, since p trusts messages originating from p . The focus of the analysis

will be on steps annotated with '+/-', which are steps 2, 3 and 6. For step 2 , the outcome is considered to be positive if p can believe that the sender is honest about the semantics of the message while for steps 3 and 6, the outcome is positive if the computations performed by p produce "positive" results.

Lets take step 2 into consideration first. A positive outcome is defined as one where the information contained in *Response* is accurate (the information source is honest). This can be expressed as:

$$p \models outcome_p = + \Leftrightarrow p \models s \overset{h}{\models} Response. \tag{5.13}$$

To determine whether the invariant holds after the execution of step 2, one can show whether $p \models s \overset{h}{\models} Response$ holds at the end of executing step 2. Using the trust inference rule, $p \models s \overset{h}{\models} Response$ can be achieved if $p \overset{h}{\models} T(s, Respond)$ and $p \models s \models Respond$. Recall that the semantic of $p \models s \models Respond$ simply refers to information origin and information timeliness. Therefore, the establishment of the condition can be achieved using authentication techniques. If the authentication layer of the trust establishment model can provide the necessary guarantees (analyzed with BAN logic), then this condition can be achieved.

In the case of a centralized protocol, an authentication between p and s can verify that *Response* comes from s and it is current. However, s is not the information origin. s provides a centralized storage for the information contained in *Response*, such as the identity of peers who have the file, and their reputation ratings. In this case, s is simply reproducing the information in the storage and therefore, $p \models s \overset{h}{\models} Response$ cannot hold, since *Response* is not in the knowledge

of s.

In the case of a decentralized protocol, responses are solicited from multiple peers in the network. In this case, s represent a union of peers who contributed to *Response*. This includes peers who are willing to share the file and also peers who provide feedback to the reputations of the responding peers. To authenticate s means to authenticate each of these peers. Therefore, suppose s consists of a set of peers $\{p_1, \ldots, p_n\}$ who contributed to *Response*, then

$$\frac{p \models T((p_1 \wedge \ldots \wedge p_n), Response), \quad p \models (p_1 \wedge \ldots \wedge p_n) \models Response}{p \models (p_1 \wedge \ldots \wedge p_n) \overset{h}{\models} Response} .$$

Assuming that the authentication layer provides authentication services (other than null authentication protocol), each of the p_i can be authenticated individually to establish $p \models (p_1 \wedge \ldots \wedge p_n) \models Response$. The remaining question is whether $p \models T((p_1 \wedge \ldots \wedge p_n), Response)$ can be established. Assuming all the peers who respond can be authenticated, the invariant holds for step 2 if p trusts all the peers in $\{p_1, \ldots, p_n\}$. Since $p \models T((p_1 \wedge \ldots \wedge p_n), Response)$ is the trust assumption of the application context, whether this can hold depends on the context. For the case of P2P file sharing, in general, p is not expected to trust every peer in the set.

For step 3, the outcome is positive if the reputation system can predict the behavior of peers accurately. The presence of an adversary algorithm presented above demonstrates that a reputation system can fail and therefore, the invariant does not hold for step 3.

Finally for step 6, a positive outcome is achieved by the successful verification

of the received file. If p requests a file f and q sends f', then

$$outcome_p = + \Leftrightarrow f = f'.$$

$Verify$ in step 6 requires the inspection of f'. Very often, this requires manual inspection of f' by going through the entire content of f' to be sure that the f' is correct. For example, if the file is a text file, p will have to read the entire file to be sure that the file is correct. Since q has the freedom of choosing any file to send to p, the invariant is not guaranteed to hold.

The above discussions address the first part of the invariant which requires the execution of all the actions to uphold a positive outcome. The second part of the invariant requires the outcome to remain positive in between the execution of all actions. Within a transaction, no other actions take place in between the actions of two steps. Therefore if $outcome_p$ is positive at t'_i, what causes $outcome_p$ to become negative at t_{i+1}? The P2P file sharing application consists of multiple transactions taking place at the same time and these transactions share global information. In the case of this model, the information is r_i, the reputation of a peer i. A transaction may make changes to information used by another transaction, resulting in a race condition. In the P2P file sharing protocol, reputation is obtained by p at t'_2 and used by p at t_3. Since the actions are not atomic, the reputation ratings obtained at t'_2 may have changed at t_3. This is potentially a problem if the peer is deemed to be trustworthy at t'_2, but at t_3, the peer's reputation falls below an acceptable threshold. In fact, an adversary can exploit the non-atomicity of the actions to delay the update of his reputation. Recall that the $Verify$ action requires manual inspection of the

entire file. Manual inspection takes time. For example, a five minutes mp3 music file requires five minutes to verify its correctness. Therefore, even when an adversary has sent an incorrect file, this attack is not discovered immediately, giving an adversary a window of opportunity to attack other peers using his outdated reputation score. Therefore, the second invariant cannot be guaranteed to hold too.

5.4 Adversarial Behavior Modeling

The adversary model describes the various behaviors of an adversary that can affect both the security of a trust establishment method and the application context. The analysis from the previous sections provide the information to model an adversary's behaviors.

The adversary algorithm, $ADV()$ constructed previously is one of the component of the model. $ADV()$ is due to the nature of reputation systems, where an adversary can accumulate sufficient reputation. The deployment of a reputation system within an application leads to more security problems. In the above example, the potential exploits of a P2P file sharing are in steps 2, 3, 6 as well as race conditions. The exploits of steps 3 and 6 are the consequences of the lack of robustness of the reputation system ($ADV()$) to select an honest peer, resulting in the downloading of an incorrect file.

The exploit of step 2 can further affect the robustness of the reputation system. This exploit is due to the possible lack of accuracy in the information that is used for trust establishment (reputation scores). Similarly, race conditions af-

Target	Behavior class
Reputation systems P2P file sharing application	Accumulate sufficient reputation.
P2P file sharing application	Provide incorrect files.
P2P file sharing application	Delay reputation updates.

Table 5.1: Adversary model for reputation systems in a P2P file sharing application.

fect the robustness of the reputation system by delaying the availability of accurate information.

Since trust establishment is dependent of the quality of the input information, the adversary model describes how an adversary can gain trust based on what an adversary can do to the inputs, rather than specific attacks. For example, the analysis for the reputation systems within a P2P file sharing can be described in Table 5.4.

In Table 5.4, the target column identifies the area where adversaries can attack. For example, accumulating sufficient reputation can be achieved by exploiting reputation systems with $ADV()$, or by causing inaccurate feedback in file sharing applications. In fact the first class of attack can be achieved by numerous methods, including malicious collectives [40], Sybil attacks [22], and so on. The second class of behavior describes an adversary's ability to send an incorrect file, regardless of his reputation. The third class of attack is to exploit the race conditions of the application. Notice that there are various methods to achieve each attack class. However, all the methods for a single class share a common behavior described

by the class. For example, $ADV()$ and malicious collectives all lead to inaccurate reputation scores. The reason for describing classes of attacks rather than specific attacks is that the list of possible attacks for a behavior class is potentially large. Moreover, there may be attacks that are currently unknown. Therefore, it is infeasible to describe all possible forms of attacks. Since all attacks of a class leads to the same consequence, it is not necessary to know the specifics. In the future, if a new form of attack is available, and if this new attack belongs to a known attack class, then the effect of this new attack is already known.

Chapter 6

Trust Establishment With Accountability

The previous chapter demonstrates the security problems of establishing trust with reputation systems in P2P file sharing. The adversary model provides an understanding of the capabilities of the adversaries. With this understanding, this chapter proceeds to describe a solution to improve the robustness of reputation systems.

It is shown that the actions of both the reputation system and the application can be exploited by adversaries to gain trust. Since these actions are essential to the proper functioning of the reputation system and the application, it is not possible to remove these actions or to restrict access to these actions. For example, it is not possible to prevent principals from building good reputations by behaving well, otherwise, it defeats the whole purpose of using reputation systems. Therefore, the design of a robust solution must not rely on inhibiting these capabilities from the participating principals. Instead, a solution has to accommodate such situations. The design philosophy behind the solution is "if something bad happens, then eventually, something good will happen." In other words, the approach to the solution is to allow recovery from attacks.

6.1 Motivation

eBay is probably one of the most well-known applications that uses a reputation system. The popularity of eBay suggests that its method of handling fraud is successful to a certain extent. Besides providing a reputation mechanism, eBay supplements its reputation systems with different protection measures, such as dispute resolution, fraud protection and even refers a fraudulent seller to law enforcement agencies. While these measures do not prevent fraud from happening, they do provide some means of recovering from a fraud. In fact, such recovery measures are very common in modern societies. For example, the policies of video rental stores are designed to accommodate failures by requiring customers to have a valid credit card so that in the event where customers do not return their rented items, their credit card accounts are charged. Such practices impose accountability on the principals involved. That is, while principals are free to do what they want, they have to answer for their own actions. When things do not go accordingly to an agreement, the errant party can be held responsible. This may include monetary compensation or even legal actions.

While accountability is widely practiced, it is a challenge to enforce in a decentralized environment, due to the lack of an authority to enforce rules. Another challenge posed by a decentralized environment is the lack of strong identities to identify principals.

However, the reality of the situation can be described by the accountability model in Fig. 6.1 where the dotted lines link a physical person to his online alter ego.

As shown in the diagram, the physical users are bounded by laws and social order. The social order corresponds to Barber's definition of trust where each member of a society is expected to carry out specific roles in a competent manner. While decentralized networks mirror an open world model where there are no authorities to monitor and control the activities of the participants, the real physical persons behind these online entities exist in a world that is governed by laws and social order. Accountability can be enforced if an online entity can be linked to a physical identity. When such linkage can be established, the principal can be held accountable for any malicious deeds, based on non-repudiable evidences collected by the authentication layer.

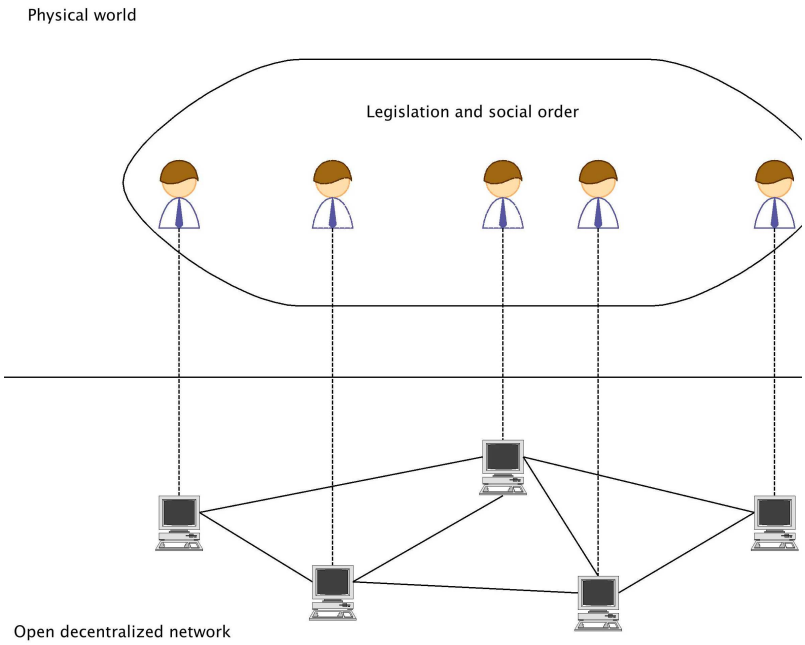

Figure 6.1: Accountability model where each online identity is related to a person in the real world.

P2P networks have a unique property that makes it relatively easy to establish

accountability. While P2P networks are often described as decentralized networks, they are actually build on top of the Internet. Therefore, it is possible to take advantage of this unique property to establish accountability. Consider a peer p who wants to download an academic paper. Besides using reputation to decide on the file provider, p may want to reduce the risk of downloading viruses by choosing to only download from peers who are researchers. The idea behind this policy is that researchers are more likely to have a correct copy of the paper, and more important, the knowledge that the file provider is a researcher can allow p to take action, such as filing a complaint to the provider's employers. To select a provider, p must have the means to verify that the provider satisfies p's policy and p must also establish a means to locate the physical identity of the principal that provides the file. A traditional approach is for a provider, q, to acquire credentials (perhaps SPKI/SDSI credentials [23]) that can attest to q's occupation and affiliation. However, a peer will have to obtain these credentials in advance and this requires the peer to predict the type of credentials that are needed for his future transactions. Alternatively, P2P users can make use of the physical network to access the massive amount of information on the web. Many Internet users have home pages, blogs or social network profiles that can serve as starting points to gather accountability information. Returning to the example, if p knows the home page of a file provider q, then based on the URL of the home page, p may established that q is affiliated with a research laboratory. However, the information does not establish the fact that q is a researcher. This may be established through other means, such as searching for q's publication track records. Should q send a correct file to p, p is able to identify q's

physical identity, locate q and the necessary authority that q answers to.

6.2 Security Analysis

To analyze the security of using accountability, the same approach from the previous chapter is used. First, the trust establishment method is analyzed on its own, without regard to the application context. Then, the deployment of the trust establishment method is taken into consideration.

The intuition behind accountability is simple, a malicious peer does not want to be accountable for his actions and will therefore attempt to hide information that allows him to be traced. On the other hand, an honest user may be willing to take responsibility for his action. This willingness to be accountable can allow the honest user to be trusted. Based on this intuition, one can distinguish honest users from malicious ones based on whether accountability can be established or not.

6.2.1 Robustness of Reputation System With Accountability

To demonstrate how accountability can improve robustness, let $\Pi_{rep}(p,q)'$ be a reputation system that has been reinforced with accountability. Once again, let C be a given context. $\Pi_{rep}(p,q)'$ can be described as follows:

1. Query C about p and q's reputation.

2. Establish p and q's accountability.

3. Output the peer who can be held accountable and who has a higher reputation, otherwise output the peer with higher reputation .

As an example, let r_p and r_q be the reputation of p and q respectively, such that $r_p > r_q$. If there is sufficient information to establish the accountability of both p and q, then $\Pi_{rep}(p,q)'$ outputs p. Similarly, if there is insufficient information to establish the accountability of both p and q, then the output is p. If accountability can only be established on one of p or q , then that principal is chosen.

Assuming p is the correct answer, let $acct_p$ represents a predicate that is true if principal p can be held accountable, and false otherwise. Since all principals make their own decisions about whether to provide accountability information, $acct_p$ and $acct_q$ are independent variables, for $p \neq q$. Also, let α be the probability that there exists information that can be used to hold a malicious peer accountable, while β is the probability that no accountability information is available for an honest peer. Since an adversary is assumed to have the ability to accumulate good reputation that is indistinguishable from the reputation of an honest principal, both r_p and r_q have an equal chance of being greater than the other. The robustness of $\Pi'_{rep}(p,q)$ is dependent on α and β:

$$\begin{aligned} Prob(\Pi_{rep}(p,q)' = p) &= Prob(acct_p \wedge acct_q \wedge r_p > r_q) \\ &\quad + Prob(acct_p \wedge \neg acct_q) \\ &\quad + Prob(\neg acct_p \wedge \neg acct_q \wedge r_p > r_q) \\ &= \frac{(1-\beta)\alpha}{2} + (1-\beta)(1-\alpha) + \frac{\beta(1-\alpha)}{2} \\ &= 1 - \frac{\alpha+\beta}{2}, \end{aligned}$$

When the sum of α and β is close to 1, then the probability of $\Pi_{rep}(p,q)'$ producing the correct answer is close to $\frac{1}{2}$, which is not robust. However, $\Pi_{rep}(p,q)'$

can be robust if the sum of α and β is small enough. In the next chapter, experiments will be described to determine the appropriate values of α and β.

The robustness of using accountability as a trust establishment method depends on two factors. First, adversaries are unwilling to be responsible for any attacks and therefore, adversaries are reluctant to associate their online identities to their physical identities. Therefore, α may be a small value. Second, since honest principals are not going to commit any malicious acts, they have little to lose by revealing some information that can be used to establish accountability. Revealing some information for identifying physical computer users may violate the privacy of the users. However, each user has the choice to determine the amount of information he or she wishes to release. Moreover, the popularity of blogs and social network profiles suggests that plenty of information regarding average Internet users is already available, and so is the willingness of Internet users to provide information about themselves. The amount of information to release is determined by their own level of comfort. For example, some users have no problem posting their photos on the web, while some may be more reserved about posting their photos. Based on this observation, β may be a small value. Therefore, these observations support the feasibility of establishing accountability on the Internet. Even without information in the form of blogs or social network profiles, one may make use of information such as domain names found on email addresses (excluding free, web based emails) or URLs to associate Internet users with some organizations.

Freeloader is a common problem in P2P networks. In the case of file sharing, there is very little incentive for a peer, q, to share a file. On top of that, q will have

to reveal some information to provide accountability so that q may be chosen. This makes the incentive to share files even lower. However, the problem of freeloading can be mitigated by providing incentives for peers to share a file. For example, in [24] the reputation of peers can affect their abilities to download a file. This provides incentives for peers to share a file and enhance their reputations so that they can download files in the future.

6.2.2 Robustness of Reputation System With Accountability in a P2P Network

The use of accountability in trust establishment serves as a "safety net," so that should a transaction go bad, there are some means to recover from it. With this in mind, the security objective has changed. In this case, the requirement is that if at time t, where $t_1 \leq t \leq t'_7$, $outcome_p = -$, then eventually at time t', such that $t' > t'_7$, $outcome_p = +$. In the event that the reputation system selects a malicious peer, if the malicious peer can be held accountable, and assuming that there exists an authority outside of the P2P network to prosecute the errant peer, then eventually, $outcome_p = +$. Similarly, if an honest peer is chosen, if the honest user can be held accountable, then eventually, $outcome_p = +$. The only situation when the condition cannot be satisfied is when accountability cannot be established.

The availability of accountability information is critical to the security of the P2P file sharing application. An adversary may pretend to be honest by providing false accountability information that belongs to another peer. To avoid this, the

authentication layer has to verify the ownership of such information. In the next section, an implementation design is laid out. Based on this design, authentication methods are suggested.

6.3 Semantic Web Enabled Trust Establishment

This section describes an implementation design to support accountability in trust establishment. Since the establishment of accountability requires some information gathering efforts, the design requires access to information from the web. Although P2P networks are logically decentralized, they can benefit from this solution since physically, P2P networks are overlaid on top of the Internet. Access to information on the web is not necessary for establishing accountability if there are sufficient information residing in decentralized networks.

In addition, this design is based on the assumption that most peers (probability of $1 - \beta$) have information on the web that can be linked to a physical identity. As previously mentioned, the success of social networks and blogs on the web suggests that this is a reasonable assumption. Even if a peer does not have any information, it is easy to create one using some of the popular social network applications.

Central to the idea of gathering information is to determine what type of information to gather. While a user can gather information by surfing the web, it is a tedious process, especially if the information available can be potentially large. To automate this process, the information has to be described in a format that can be processed by a computer. The natural solution to this is to use semantic web

ontology to markup the information.

6.3.1 Identity

The popularity of blogs and social network sites, such as MySpace or Facebook, means that there is plenty of information about a person on the web. A convenient way to link an online pseudonym to a physical identity is to use the Uniform Resource Identifier(URI) of an online profile as the online identity. Since a pseudonym is a string of characters, an URI can be used as the pseudonym of a peer in P2P networks. If a user does not have an existing profile, it is also easy to create one. Therefore, the use of URIs as pseudonyms can be easily adopted without additional software or hardware. For simplicity, the focus of this dissertation will be on social network profiles. The same line of reasoning can be applied to blogs or home pages. Moreover, the use of social network profiles has an additional advantage of tapping into existing trust relationships among Internet users which is based on friendship. These trust relationships can be used as a basis for the trust assumptions ($p \models T(q, x)$) presented in the previous chapter.

While there are numerous social network sites available, the use of semantic web means that the Friend-of-a-Friend (FOAF)[1] is the natural format for representing social network profiles. FOAF is an ontology based on the semantic web ontology language, OWL. The FOAF ontology models a social network by describing people, organizations and their relationships. For example, the *foaf:Person* class describes a person while the property, *foaf:name*, describes a person's name. The

[1]FOAF Project: http://www.foaf-project.org

foaf:knows relationship can be used to express friendship. In the syntax of OWL, the term after the colon identifies a class (*Person*) or a property of a class (*name*). These terms are defined in the name space indicated before the colon (*foaf*). For more information on the FOAF ontology, please refer to [13]. Since FOAF is based on OWL, one can take advantage of the extensibility of OWL to enhance the basic FOAF ontology with domain-specific knowledge.

Creating a FOAF profile is easy, as it does not require knowledge about OWL. To create a profile, simply go to the FOAF project web site to fill up an online form. The OWL code will be generated automatically. This code can then be copied and published on the Internet. Unlike other popular social network applications, FOAF does not use a central storage for storing FOAF profiles. It is up to a user to decide the location to publish his own profile. Moreover, the information belongs to the users and any machine capable of understanding OWL can process the FOAF profile. In comparison, the information in many social network applications belongs to the application proprietor and cannot be exported in machine processable format legally.

To link FOAF profiles to peers in P2P networks, the URI where a FOAF profile is hosted is used as a peer's pseudonym. A person may have multiple FOAF profiles. It is up to the person to decide which profile is to be associated with the pseudonym. For example, a person may use a profile that describes his research activities when sharing research papers, while a profile which describes the social activities of the same person can be used for sharing files of other nature. The OWL language allows two FOAF profiles to be linked to the same person by using

the *owl:sameAs* property. A person with multiple FOAF profiles may make use of this to link all profiles belonging to him, allowing a broader search for information. However, this practice of linking multiple profiles is optional and is completely up to the individual.

The concept of using identity to enforce accountability shares a common feature with the OpenID project, [2] which is the use of a URI as identity. However, the motivation for using a URI as identity is different in these two schemes. For the OpenID project, the primary motivation is to provide convenience to authenticate users by using a single identity and a single password to log onto various web sites. In addition, it also allows a user to prove ownership of the web site pointed to by the identity. For this work, the motivation for using URI as identity is to facilitate the gathering of information that can be used to achieve accountability. Therefore, an OpenID can be used as an identity for trust establishment if the URI points to information that can allow one to establish accountability. In fact, the latest FOAF ontology supports the use of OpenID.

The use of information to establish accountability may cause concerns about the privacy of users. However, privacy and trust are often in conflict where the establishment of trust in a principal requires the release of knowledge about the principal, which results in the loss of privacy [65]. In this case, users give up some privacy in exchange for trust. In this work, the owner of a profile has full control of what information to be made available. Therefore, the owner can decide on the amount of information to release. The use of a social network profile for information

[2]OpenID: http://openid.net"

gathering is non-invasive in the sense that the only information gathered are those that are already available in the web. However, this does not prevent a principal from publishing information regarding another principal on his own site.

6.3.2 Authentication

Since anyone can claim any URI as identity, it is necessary to authenticate the ownership of a URI. When a URI is presented as a pseudonym, the authentication layer has to authenticate the ownership of the URI. The most obvious way is through cryptographic means. The Web-of-Trust (WOT) Ontology [3] allows one to attach digital signatures on semantic web documents to prove ownership of the documents. In addition, this also allows one to verify that the document has not been modified. A disadvantage of this approach is that the verifier is made to verify digital signatures, which is a relatively expensive operation. This may allow an adversary to carry out denial-of-access attack by presenting many URIs for the verifier to verify.

The extensibility of the OWL language means that the FOAF ontology can be extended to describe the cryptographic keys of the owner of the profile. This allows the profile owner to attach his public key to the profile. By doing so, a standard challenge and response authentication protocol can be used to prove that a peer is the owner of a FOAF profile and also to establish a secure channel for file transfer.

A problem with using a PKI is to establish trust in the public key. For example, an adversary can duplicate the contents of another profile. To claim ownership

[3]WOT Ontology: http://xmlns.com/wot/0.1/

to the duplicated profile, the adversary replaces the public key in the duplicated profile with his own. The adversary can then sign the duplicated document using the corresponding signature key. Therefore, a certain degree of trust in the public key has to be established in order for authentication to be meaningful. If the PKI involved is based on a centralized model, the digital certificate issued by a CA is used to establish trust in the public key. On the other hand, if a decentralized PKI is used, then the trust in the public key can be based on the web-of-trust model.

6.3.3 Provenance

If the public key of the owner of a profile is trusted, then the contents of the profile may be trusted if it is signed by the corresponding private key. However, this may not be sufficient in some cases, for example, if sufficient trust in the key cannot be derived. In this case, some additional measures have to be taken.

To illustrate, consider the following example. Suppose a peer is identified by the URI, "http://www.jdol.net/JDole.owl". This URI links to a resource with FOAF properties, *foaf:workplaceHomepage* and *foaf:workInfoHomepage*, taking the values of "www.cs.foo.edu", and "www.cs.foo.edu/ jdole" respectively. The first property, *foaf:workplaceHomepage*, identifies the website of the organization where the peer is employed, while the *foaf:workInfoHomepage* is the website that contains information about the peer's work. Given these statements, how can one verify their correctness?

The peer, JDole, can prove the claims in JDole.owl by using digital credentials.

For example, the employer can issue a signed credential to certify that someone is an employee. However, a machine is unlikely to be able to process such credentials as they are designed primarily for humans to read. An obvious solution would be to express such credentials in a semantic web ontology language (OWL). These OWL statements can then be signed by the issuer. In this arrangement, the employer is treated as a credential issuer, certifying the statements. In the above example, the two properties *foaf:workplaceHomepage* and *foaf:workInfoHomepage* can be signed by the organization identified by the URI "www.cs.foo.edu". Trust in the employer's signature key can be established by the employer's public key certificate. This approach is particularly useful if the issuer is a well-established organization, such as government agencies, universities or corporations with well-established brand names. A drawback with this approach is that a peer will have to obtain such credentials in advance.

The above example can be implemented as shown by the snippets of OWL statements in Fig. 6.2. Lines 1-8 are the contents of a file which describe the work place information home page of a person. Also included in this block of statements are annotations which identify the authority who can vouch for the truth of the statement (line 5), link to the certificate of the authority in line 6 and the signature of the authority (line 7). A verifier can obtain the public key by following the link in line 6 to the certificate and verify the signature as pointed by the URI in line 7. A peer can prove to be the subject of the statement by demonstrating the knowledge of the secret key that corresponds to the public key found in the certificate mentioned in line 8. The properties in lines 5-8 are defined in the ontology in the "cred" namespace

(identified by the fictitious URI, "http://www.credentials.net/Credentials.owl"). In this ontology, these properties are defined as annotation properties and snippets of the ontology are described in lines 9-11.

The definition of "hasAuthorityCertificate" states that for line 6 to be valid, the resource pointed by the URI, "http://www.cs.foo,edu/FOO.owl#FOOCertificate" must be an instance of "X.509Certificate". In line 11, the notation "#X.509Certificate" implies that the definition of "#X.509Certificate" can be located in the current file. However, its definition is left out in this example.

6.3.4 Information Gathering

A useful piece of information for establishing accountability is one that allows the owner of the profile to be traced. The basic FOAF ontology allows a user to describe both the work place home page and the school home page, which can be used to establish the locality of the user. This can act as a deterrent to prevent malicious users from attacking since they can be located. In the event of an attack, the victim can turn to local law enforcement for help. In addition, the FOAF ontology can be extended to allow victims to announce attacks. For example, one can define a "$victimOf$" relationship between two *foaf:Person*. If p is a victim of q's attack, p can announce that he is a victim using the $victimOf$ property.

There exists different forms of social networks on the web. For example, the FOAF is one social network, while an online discussion forum is another social network. An Internet user may participate in more than one social network, allowing

Contents of http://www.jdol.net/JDole.owl:

```
<rdf:RDF>
1: xmlns:foaf="http://xmlns.com/foaf/0.1/#"
2: xmlns:cred="http://www.credentials.net/Credentials.owl#"
                  :
                  :
3: <foaf:Person rdf:ID="jdole">
4:  <foaf:workInfoHomepage rdf:resource=
             "http://www.cs.foo.edu/~jdole"/>
5:  <cred:hasAuthority rdf:resource="http://www.cs.foo.edu"/>
6:  <cred:hasAuthorityCertificate rdf:resource=
          "http://www.cs.foo.edu/FOO.owl#FOOCertificate"/>
7:  <cred:hasAuthoritySignature
         rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
              "http://www.jdol.net/JDole.sig"
    </cred:hasAuthoritySignature>
8:  <cred:hasProof rdf:resource="#jdoleCertificate"/>
   </foaf:Person>
</rdf:RDF>
```

Contents of http://www.credentials.net/Credentials.owl:

```
9: <owl:AnnotationProperty rdf:ID="hasAuthorityCertificate">
10:  <rdf:type rdf:resource=
              "http://www.w3.org/2002/07/owl#ObjectProperty"/>
11:  <rdfs:range rdf:resource="#X.509Certificate"/>
   </owl:AnnotationProperty>
```

Figure 6.2: Sample OWL statements describing contents of a peer's FOAF profile and the "hasAuthorityCertificate" property.

the different networks to be "stitched" together. For example, if semantic web research is the subject of interest, suppose that q is a potential provider. If q's FOAF profile provides a link to q's profile, which allows someone to establish q's contribution in the semantic web research community, some form of accountability

can be achieved. For example, Flink, which is a social network of semantic web researcher, provides information such as the research contributions and co-author relationships within the semantic web research community. If q's FOAF profile is linked to his Flink profile, one may establish q's influence within the research community. If q is well known within the community, locating q is easy within the community. Moreover, if q is well respected, q is not likely to risk losing his real life reputation by carrying out malicious acts. Therefore, these simple observations can increase the confidence in making trust decisions. Moreover, it is more difficult for an adversary to gain trust, since the adversary must make significant contributions in the semantic web research community before he is trusted.

The method of establishing accountability is a passive one that relies on peers to use URIs to their profiles as identity and to decide on how much information to release publicly. There may be situations where the profile owner does not release sufficient information to establish accountability. In this case, an active information gathering can be done. This involves using search engines, such as Google, or Swoogle, which is a semantic web search engine to search for relevant information. For example, if p is a victim of q, this information is not likely to be found in q's profile, but p may publish it somewhere in the web. This information may be retrieved by using a search engine. A problem with using a search engine is that a search may return a potentially large set of results. It may be computationally intensive to look through every one of them. Of course, p may bear some grudges against q and spread lies to hurt q. Therefore, the truthfulness of information still has to undergo verification.

Chapter 7

Empirical Analysis

To validate the solutions presented in the previous chapter, simulations are designed to evaluate the effectiveness of adding accountability to reputation systems. The experiments also aim to study the effects of environment variables: the ratio of good and bad peers, update delays, α and β which determine the availability of accountability information.

7.1 Experiment Setup

The simulations are conducted on the Query Cycle Simulator [63]. This simulator provides a platform for simulating file sharing in a P2P network. The Query Cycle Simulator uses the EigenTrust algorithm for computing the reputation of peers in the network. Since the primary objective is to demonstrate that accountability can improve reputation systems, the implementation of the EigenTrust algorithm is a useful feature of the Query Cycle Simulator.

The basic simulation setup consists of 100 good peers and 20 different file categories. The files are distributed according to the distribution model in [62], with each peer being interested in a set of randomly chosen categories.

The simulation consists of many cycles. In a single cycle, the peers will take turns selecting a category of interest randomly, followed by choosing a file from that category randomly. If the peer already has the file, the peer will select another file. Otherwise, the peer queries the network for the file. Each query message is broadcasted to the neighboring peers who will forward the message to their neighbors. The forwarding is controlled by the Time To Live (TTL) value of the query message. Each time the message is forwarded, the value is decreased by 1. When the value reaches 0, the message is dropped. The default value of 3 is used in the simulations.

File requesting peers use the EigenTrust reputation system to select file providers. The simulator provides the following selection strategies: random selection, global trust deterministic, global trust probabilistic, local trust probabilistic and query response time. The random selection and query response time strategies do not take reputation into account. As the name implies, random selection picks a provider randomly. The query response time method selects the peer whose response is first received. Under the EigenTrust reputation scheme, each peer maintains a local reputation table of peers with whom he has previously interacted. The EigenTrust algorithm combines all local tables to form a global reputation table. The local trust selection strategy chooses a peer based on the local reputation table while the global selection strategy uses the global table. The global and local trust selection methods can be either probabilistic or deterministic. The probabilistic method assigns a probability to each responding peer based on their reputation. Based on this probability distribution, a provider is randomly chosen. Using the probabilistic method, even peers with low or no reputation have a chance of being selected. On

the other hand, the deterministic method simply selects the peer with the highest reputation. In the experiments, the global trust deterministic method is used as the selection strategy.

In this experiment, the following selection strategies are used: global trust deterministic and random selection. The performances of these two stragegies will be compared to the performance of EigenTrust with accountability. The global trust deterministic simulates the reputation system modeled in the previous chapters. Recall that in the definition of robustness, the performance of the random selection strategy establishes the baseline on whether a trust establishment is robust or not. Therefore, the use of random strategy in this experiment offers this baseline on the experiment results: a trust establishment method is not sufficiently robust if it performs worse than the random strategy.

Once the provider is selected, the requesting peer will request a download. An honest provider will provide a correct file, while a malicious provider will provide an incorrect file. If the download attempt fails (incorrect file), the requesting peer will attempt to download from the next peer in the list of responding peers. This continues until either the download is successful, or the list of providers is exhausted. Based on the outcome, the requesting peer will update the trust value of the providers using the EigenTrust algorithm.

Since the interest of this research is the robustness of trust establishment, the performance of trust establishment methods can be based on their robustness. Recall that robustness is a measure of the likelihood of success in distinguishing between honest and malicious peers. This can be estimated by the total number of

bad downloads over the total downloads per cycle for all peers:

$$\rho = \frac{\text{total bad downloads}}{\text{total bad downloads} + \text{total good downloads}}. \tag{7.1}$$

Let this ratio be denoted by ρ. The ratio, ρ measures the percentage of downloads that are bad. Therefore the trust establishment is less robust for higher values of ρ.

The performance indicator, ρ, is affected by the number of malicious peers. In an ideal case with no malicious peers, ρ should be 0 or close to 0 in the presence of errors. To model how the number of malicious peers affect the performance, malicious peers are introduced into the experiment in incremental steps of 10, while keeping the number of good peers at 100. By keeping the number of good peers at 100, one can ensure that there are sufficient good peers around to share files. The number of malicious nodes is represented by μ.

While ρ measures robustness, it does not measure effectiveness. For example if a trust establishment method results in 100 bad downloads while another method results in 1000 bad downloads with all other parameters being equal. Also, suppose that both methods have a bad download rate of 0.5. Both methods are equally robust. However, the first method is more effective since it only results in 100 bad downloads. To measure the effectiveness of a trust establishment method, let σ be the total number of bad downloads in a single cycle.

To study the effects of enhancing reputation systems with accountability, modifications are made to the simulator as follows. When a peer is presented with a list of providers, the peer will make three selections. The first selection is based on picking a peer using the global deterministic strategy, which only considers reputation.

The second selection method picks the peer with the highest reputation and who has accountability information. Peers whose accountability cannot be established are dropped. The ability to establish accountability is controlled by two parameters. The parameter, α, determines the percentage of malicious peers with accountability information. Based on the value of α, malicious peers are tagged at the beginning of the simulation to indicate whether they have accountability information or not. The second parameter, β, determines the percentage of good peers with accountability information. Similarly, the good peers are tagged to indicate whether they have accountability information or not. To model the effect where an adversary can delay the update of reputation, the variable δ represents such delay in terms of number of cycles. The third selection is to choose a provider randomly.

7.2 Simulation Scenarios

The simulation scenarios are designed to compare the robustness of the four trust establishment methods mentioned earlier when they are attacked by adversaries. The adversary model from Chapter 5 provides an insight to the type of attacks to consider when designing the scenarios.

The environment variables are μ, α, β and δ. The scenarios will study how these variables affect the robustness of the trust establishment methods.

Designing simulations can be a challenge. Many research efforts demonstrate the effectiveness of their trust establishment methods by simulating against various attack scenarios. As mentioned in Chapter 3 many of these works do not justify that

their choices of attack scenarios are adequate. For example, the EigenTrust simulation scenarios focus on attackers who form cooperating collectives. However, while such choices are valid, they are not complete. The analysis in the Chapter 5 reveals other types of attacks that have not been considered in the EigenTrust research, such as the race conditions and the adversary's ability to accumulate sufficient reputation.

In fact, one major difference between the simulations in this dissertation and those in EigenTrust research [40] is regarding the initial reputation of the adversaries. In this dissertation, it is assumed that adversaries have already established sufficient reputation at the beginning of the simulations, while in the EigenTrust simulations, the adversaries are initialized with neutral reputation scores. The reason for this difference is because in Chapter 5, it has been established that an adversary has the capability to accumulate sufficient reputation. Therefore, the interest is to study how this attack class affects the robustness of the trust establishment methods. On the other hand, the EigenTrust research focus is to show that it is difficult for an adversary to gain trust using specific attack methods. However, all they have achieved is to show that those chosen attacks cannot allow an adversary to build a good reputation successfully. Unfortunately, they neglect to consider other methods in which adversaries can gain trust. Therefore, it is unreasonable to claim that EigenTrust is secure based on their simulation results.

An adversary model can provide insights into how the simulation scenarios can be designed. Each attack class describes what the adversaries are capable of. Therefore, a simulation scenario can be assigned to each attack class. The adversary

Target	Attack class
$\Pi'_{EigenTrust}$ P2P file sharing application	Obtain good reputation.
P2P file sharing application	Provide incorrect files.
P2P file sharing application	Delay reputation updates.
P2P file sharing application	Control α.
P2P file sharing application	Control β.

Table 7.1: Adversary model for the EigenTrust reputation system with accountability ($\Pi'_{EigenTrust}$) in a P2P file sharing application.

model for the EigenTrust reputation system ($\Pi_{EigenTrust}$) is similar to the adversary model in Table 5.4.

For the case of EigenTrust with accountability ($\Pi'_{EigenTrust}$), the robustness of this method is determined by α and β. Assuming an adversary can control these variables, then the corresponding adversary model can be described in Table 7.2.

Based on the adversary models in Table 5.4 and Table 7.2, all peers are initiated with good reputation scores at the beginning of every simulation. This arrangement simulates the first attack class in both models. In this way, the robustness of the trust establishment method under such attacks can be observed immediately, rather than waiting for the malicious peers to accumulate sufficient reputations. In reality, not all malicious peers will accumulate sufficient reputation before attacking. In the simulations, all malicious peers have accumulated sufficient reputations. This provides a worst-case scenario to test the robustness of the trust establishment methods.

To increase the likelihood of encountering malicious peers, the Query Cycle simulator inserts malicious peers such that they are neighbors to highly connected honest peers. To further increase this likelihood, malicious peers are allocated an average of five neighbors, as opposed to an average of three neighbors for honest peers.

In addition, the original EigenTrust reputation system is based on the PageRank algorithm by Google. The PageRank algorithm ranks web pages according to their importance, based on the number of links. Similarly, the EigenTrust algorithm identifies a set of peers who are deemed to be highly trustworthy and these highly trustworthy peers have more links to other peers, compared to other ordinary peers. In this case, these peers have a higher probability of being chosen and can improve the performance of EigenTrust in terms of reducing the number of inauthentic downloads. In the simulations, there are no highly trustworthy peers to give malicious peers a higher chance of being selected.

Since malicious peers do not have to accumulate sufficient reputation before attacking, a malicious peer will attack once it has been chosen as a file provider. The attack comes in the form of sending an incorrect file. An incorrect file is any file that is not requested by the requesting peer. It may be a file of different content, an incomplete file or even an executable file containing viruses or malicious software. The file may also be a "NULL" file to represent the situation where no files are sent.

The above discussion forms the basic attacks used in all simulation scenarios. Additional scenarios are derived based on the environment variables: μ, α, β and δ. The first scenario is to investigate how the ratio of good to bad peers, μ, affects the

performance of the two trust establishment methods: EigenTrust and EigenTrust with accountability.

The second scenario investigates how different values of α and β affects performance. These two variables model the likelihood of establishing accountability. In order to attack, an adversary needs to establish accountability. However, an adversary does not want to be accountable for his actions and may resort to provide false information on his profile, or even hijack the profile of another person to present it as his own. The variable α addresses the above concerns. It controls the number of adversaries that can convince a verifying peer that accountability information is present. Similarly, a malicious peer may attempt to remove accountability information of other good peers to deny services. In this case, the variable β simulates the amount of good peers who do not have accountability information. It is also possible that a good peer may not have sufficient information to establish accountability or the peer does not wish to disclose too much information.

Finally, the third scenario is to investigate the effects of exploiting the race condition within the P2P protocol to create delays (δ) in the update of reputation.

7.3 Simulation Results

Simulations are carried out with the scenarios mentioned in the previous section. A simulation run consists of 100 cycles. Therefore, each peer has to download 100 files in each simulation run. At the beginning of each run, the network topology is generated randomly, based on the Power Law distribution [50]. At the end of

each cycle, the EigenTrust algorithm is run to compute the global reputation of peers based on the results of previous cycles. Within a simulation run, the variables and topology of the network remain unchanged. Since the robustness indicator, ρ, is defined as the ratio of bad downloads to total downloads in a single cycle and a single run consists of 100 cycles, let $\overline{\rho}$ be the average of the values of ρ in a single run. Similarly, define $\overline{\sigma}$ to be the average number of bad downloads in a single run. For each scenario, a simulation run is repeated until the results converged. Let $\overline{\rho_i}$ be the result of the i^{th} run and let A_i be the average of the i runs:

$$A_i = \frac{\overline{\rho_1} + \ldots + \overline{\rho_i}}{i}. \tag{7.2}$$

Then convergence occurs at the end of the i^th run when:

$$\frac{|A_i - A_{i-1}|}{A_{i-1}} < 0.05. \tag{7.3}$$

7.3.1 How μ affects performance

The first scenario studies how the ratio of good to malicious peers affect the performance, as measured by the average effectiveness per run ($\overline{\sigma}$) and average robustness per run ($\overline{\rho}$). The number of good peers remain at 100, while the number of malicious peers, μ is initially set to 10. The value of μ is incremented in steps of 10 until μ reaches 120. For each value of μ, simulation runs (100 cycles per run) are repeated until the results converge. The values of α, β and δ are kept constant at 0.05, 0.05 and 0 respectively.

Fig. 7.1 shows the effectiveness between the four trust establishment method. The use of accountability can greatly reduces the number of inauthentic downloads.

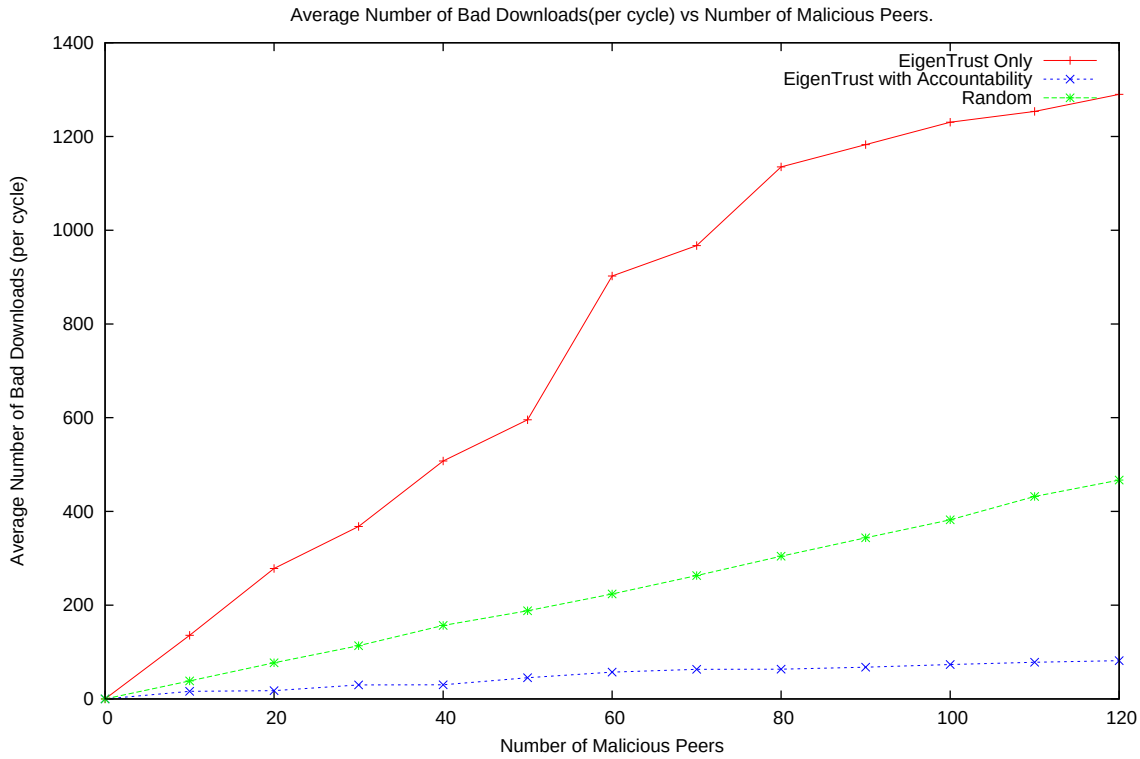

Figure 7.1: Plot showing how the number of malicious peers affects effectiveness.

Without accountability, the addition of malicious peers increases the number of bad downloads significantly. This is not surprising given that the more the number of malicious peers, the more the number of bad downloads is expected. Using random selection as the baseline, the results show that when the malicious peers manage to establish good reputations, the EigenTrust reputation system is less effective than random selection. The rate of increase in bad downloads for EigenTrust with accountability is slower. This is because the accountability method uses the availability of accountability information to make decision. Therefore, effectiveness is not determined alone by the number of malicious peers, but by other variables, including

α and β. This scenario demonstrates the potential of using accountability as a trust establishment method, provided that α and β values are in the "appropriate" range. Even when $\mu > 100$, where the number of malicious peers outnumbered the number of honest peers, the EigenTrust with accountability method is still significantly more effective than EigenTrust alone.

As for robustness, the results are shown in Fig. 7.2.

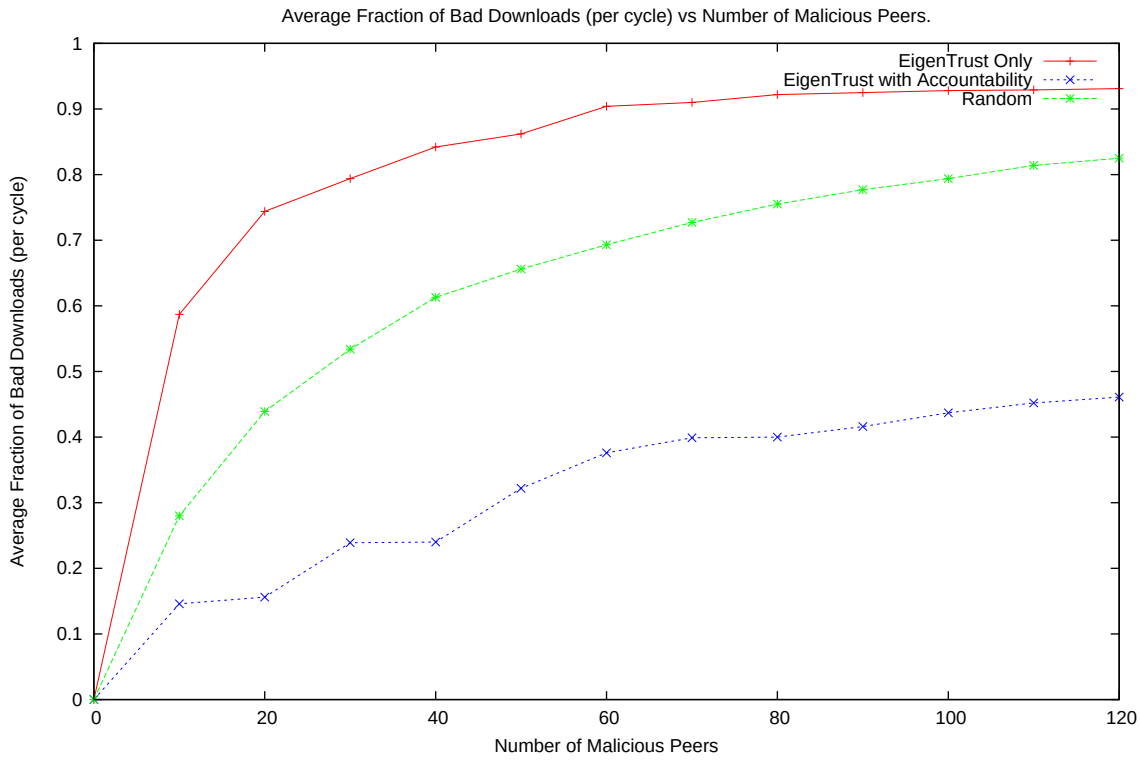

Figure 7.2: Plot showing how the number of malicious peers affects robustness.

The results show that accountability can improve the robustness of the EigenTrust reputation system significantly. When 10 malicious peers with good reputations are introduced into the network, it results in more than half of the downloads

being incorrect files if only EigenTrust is used. In this case, the reputation system alone is not robust even when there are relatively few malicious peers (ratio of malicious to honest peers is $\frac{1}{10}$. Using the random selection as the experiment control, the EigenTrust reputation system is not robust since it performs worse than the random method. On the other hand, when accountability is used the fraction of bad download is less than half, even when there are 100 malicious peers.

Referring to Equation 7.1, if the number of bad downloads is significantly greater than the number of good downloads, ρ approaches to 1. Fig. 7.2 shows that the EigenTrust method approaches this limit quickly, when the number of malicious peers reaches 60. Referring to Fig. 7.1, the number of bad downloads increases very quickly for the EigenTrust method as compared to the EigenTrust with accountability method. Therefore, the robustness of the EigenTrust with accountability does not reach the limit as quickly the EigenTrust method. In fact, with the value of μ at 120, ρ is only 0.512.

The EigenTrust method is dependent on μ while the EigenTrust with accountability method depends not only on μ but α and β as well. Therefore, the EigenTrust method is more sensitive to changes in μ as compared to EigenTrust with accountability.

7.3.2 How α and β affects performance

The next few scenarios aim to observe how trust establishment with accountability is affected by the α and β. Firstly, the influence of α is studied by varying

the value of α in a series of simulations. In these simulations, the values of β and δ are kept constant at 0.05 and 0 respectively. Different levels of malicious activities are controlled by setting μ to 10, 30 and 50. The results are presented in Figs. 7.3 and 7.4. Since the EigenTrust only method and random selection are not affected by the values of α, the results for these methods not expected to vary much.

Consider the case when α is 1. In this case, accountability can be established for all malicious peers. With β kept at a small value of 0.05, the set of responding peers being considered for the EigenTrust with accountability method will almost be the same as the set of responding peers obtained from EigenTrust only method. The only difference will be due to a small number of honest peers whose accountability cannot be established. These honest peers are less likely to be selected since EigenTrust with accountability method gives greater priority to peers (even malicious ones) who can establish accountability. In this case, the EigenTrust with accountability method may perform worse than the EigenTrust only method, due to less choices of honest peers. This is confirmed by the observation that in most cases, the EigenTrust with accountability method performs better than EigenTrust only in terms of effectiveness and robustness, but when α is around 0.85, EigenTrust only outperforms EigenTrust with accountability. The experiments suggest that if less than 85% of the malicious peers can appear to be accountable, accountability can be useful for trust establishment. To obtain better results, the simulations suggest that when the value of α is less than 0.5, significant improvement in performance can be achieved by using accountability.

When comparing the performance of EigenTrust with accountability with ran-

dom selection, EigenTrust with accountability only performs better than random selection for values of α that are less than approximately 25%. This suggests that a decentralize network that uses accountability to establish trust can accommodate about 25% of malicious peers who has accountability information.

The number of malicious peers affect the sensitivity to changes in α for both performance metrics. When there are more malicious peers, changes in α results in bigger changes in performance. This is due to the number of malicious peers who can gain trust in the network. For example, if μ is 10, even if α is 0.5, there are only 5 malicious peers who can gain trust while if μ is 50, then α only has to be as small as 0.1 to result in 5 malicious peers who can gain trust.

The next series of simulations study the effect of β on the performance of EigenTrust with accountability. The set-up is similar to the previous simulations, except that this time, α is kept at a constant value of 0.05 while the value of β varies. Like before, the number of adversary, μ, is set at 10, 30 and 50.

The effect of different values of β are shown in Figs. 7.5 and 7.6. From Fig. 7.5, the results suggest that EigenTrust with accountability performs very well in terms of the number of bad downloads. However, the study of Fig. 7.6 reveals a clearer picture. At $\beta = 1$, the fraction of bad downloads is 1. This can be explained by Equation 7.1. When all honest peers do not offer any accountability information, they are not considered for downloads. Therefore, only offers from malicious peers (who managed to present accountability information) are considered. As a result, all the download attempts lead to bad downloads. With no good downloads, the value of ρ is 1. The number of bad downloads is kept at a relatively low level for

EigenTrust with accountability due to α being assigned a low value of 0.05.

This set of experiments reveals an insight to the influence of α and β. The value of α has an impact on the effectiveness and robustness of trust establishment methods. On the other hand, β affects the availability of files. When β gets large, there are less files available for download. The experiment results suggest that EigenTrust with accountability performs reasonably well if β is approximately less than 0.75.

To observe the combination of various values of α and β, simulations are carried out with different values of μ at 10, 30 and 50. The results are shown in Figs. 7.7 and 7.8. The results suggest that EigenTrust with accountability will perform better than EigenTrust only if the α and β values are below 0.5. A significant improvement in performance can be achieved when both α and β are below 0.3.

7.3.3 How δ affects performance

To investigate the effects of delay in the updates of reputation, the simulations are performed with the number of adversaries set to 10, α and β both having the value of 0.05 for the EigenTrust only method. The delay, δ is set to 10 runs. After the 10 runs, the reputation updates for these 10 runs are performed before the rest of the runs are resumed. It is expected that the number of bad downloads do not vary much during the period when there is no update of reputation. The results are shown in Fig 7.9.

The simulation results confirmed the expectations. There is a drop in the

number of bad files once there is no delay in the updates of reputation. For comparison, the results for EigenTrust with accountability are also plotted. Even the EigenTrust with accountability method is affected by the delays. This is due to some adversaries having the ability to present accountability information. However, since the number of bad downloads for EigenTrust with accountability is much lower than EigenTrust alone, even when there is a delay in updating the reputation of peers, the number of bad downloads is still significantly lower when compared to EigenTrust only method.

7.4 Findings

The simulations reveal the promise of using accountability to reinforce reputation systems. In terms of effectiveness, using accountability manages to reduce the number of bad downloads, improving the overall effectiveness of a P2P file sharing application. Adding accountability to a reputation system also improves the robustness of reputation systems.

The use of accountability for trust establishment also provides a "safety net" feature that is not shown by the simulations. For reputation systems in a decentralized network, when bad download occurs, the victim is unable to prosecute the adversary since the identity of the adversary is hidden behind a pseudonym. The lack of centralized authorities means that there is no means of linking a pseudonym to a physical identity. However, the use of accountability attempts to establish a link between an online identity to a physical identity. Therefore, even in the absence

of a centralized authority, a victim is able to redress the situation. This ability acts as a "safety net" for the victim. In addition, it also acts as a deterrent to discourage malicious activities, since a rational adversary will not want to be held accountable for their actions.

However, it is anticipated that there are adversaries who can establish accountability, such as, by hijacking a legitimate identity. The simulations are designed with this in mind and they have demonstrated the level of robustness of using accountability in the presence of adversarial activities. In general, if μ is a small value, the use of accountability is able to achieve very good performance in terms of effectiveness and robustness, even if α and β are relatively higher values. In practice, it is expected that the value of μ is low. This is because if μ is too high, the presence of large amount of malicious activities will discourage peers from participating in file sharing and the P2P file sharing application will cease to be useful. Therefore, the use of accountability presents a useful method for trust establishment.

Finally, note that the adversary model for reputation systems with accountability has more attack classes than the adversary model for reputation systems only. However, the use of accountability results in more robustness. This shows that the robustness of trust establishment cannot be judged by the number of attack classes in the adversary model. Instead, it is the resilience of the trust establishment method against the attack classes that determine its robustness.

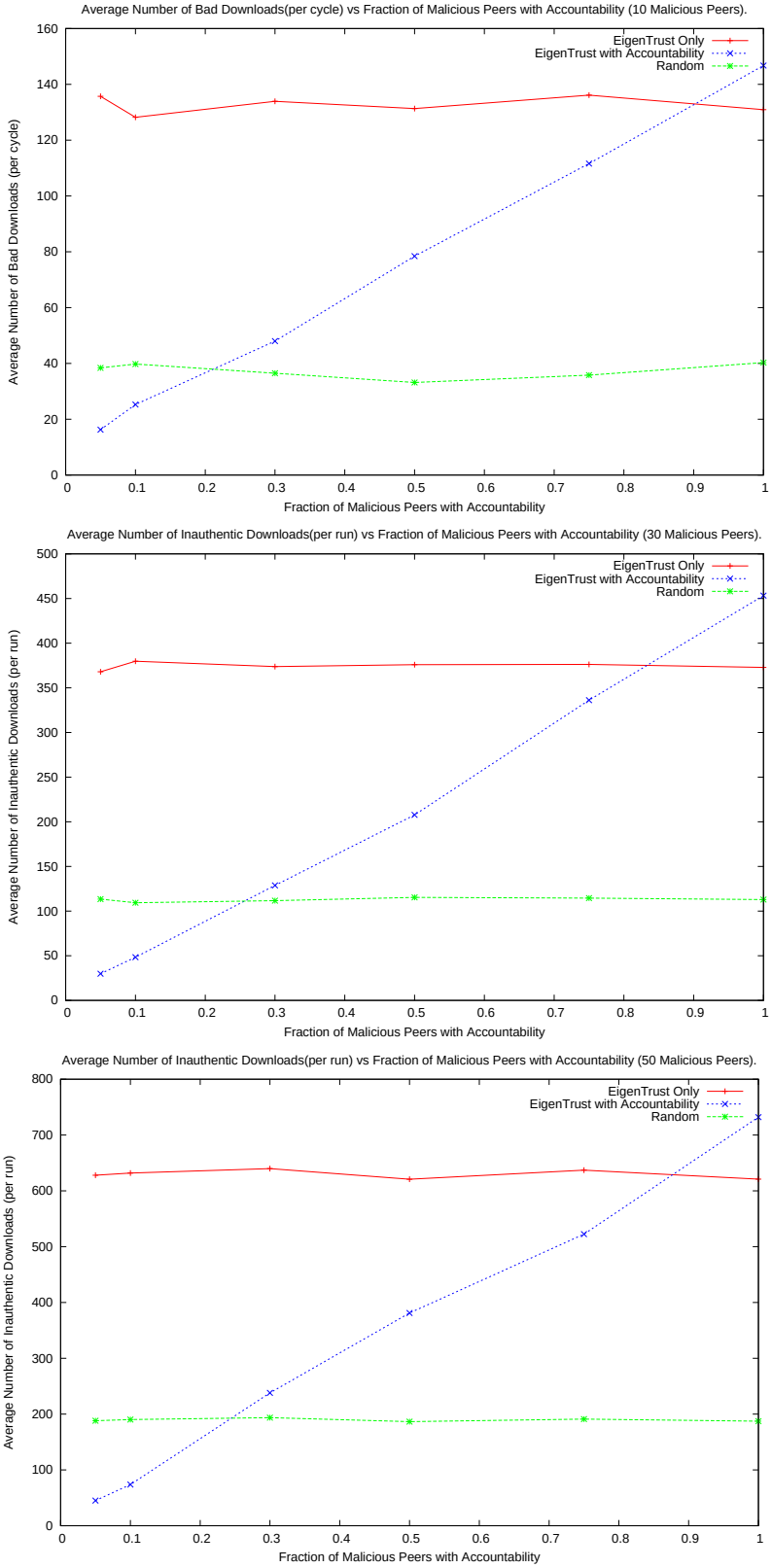

Figure 7.3: Plot showing how α affects effectiveness.

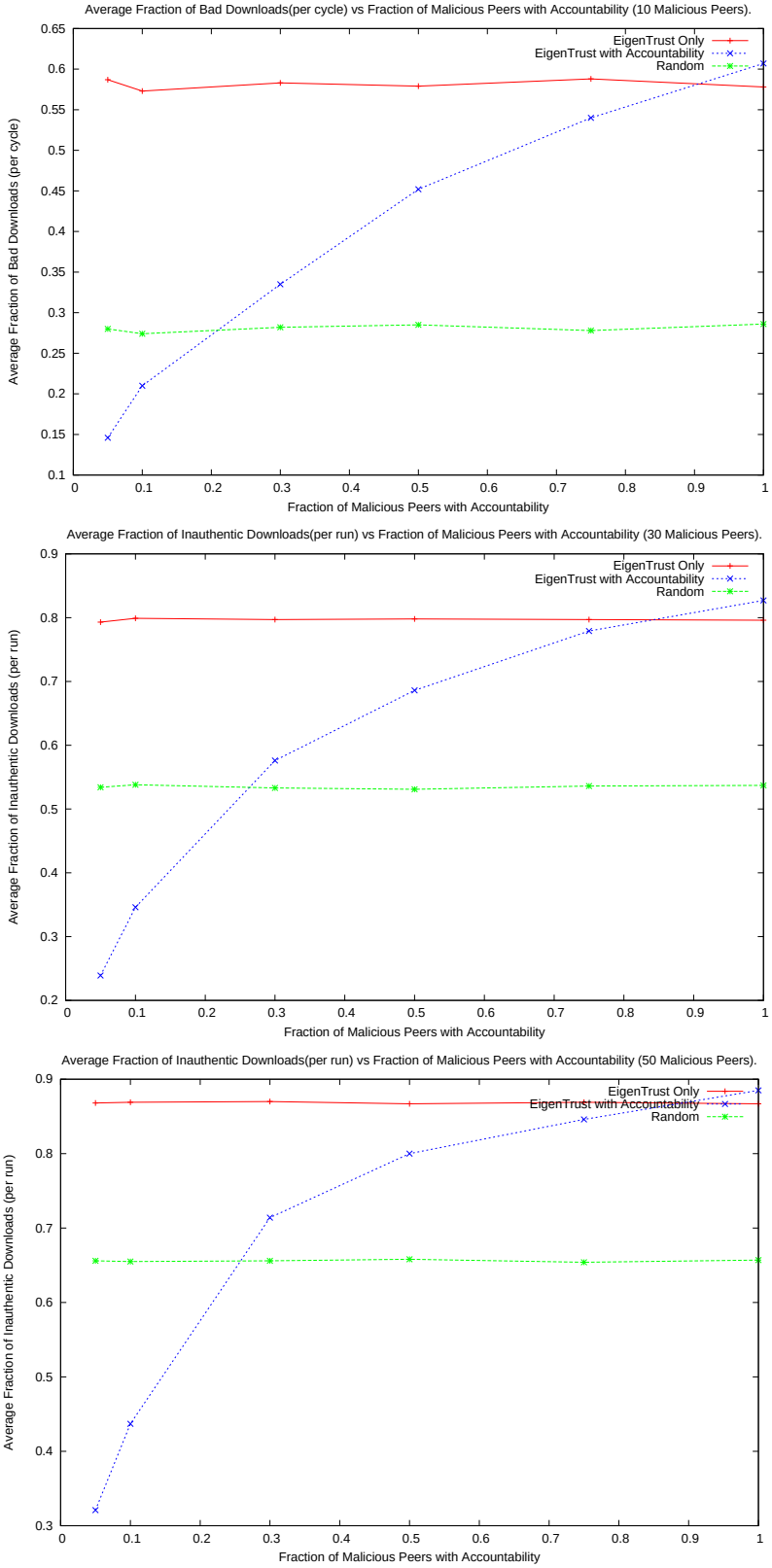

Figure 7.4: Plot showing how α affects robustness.

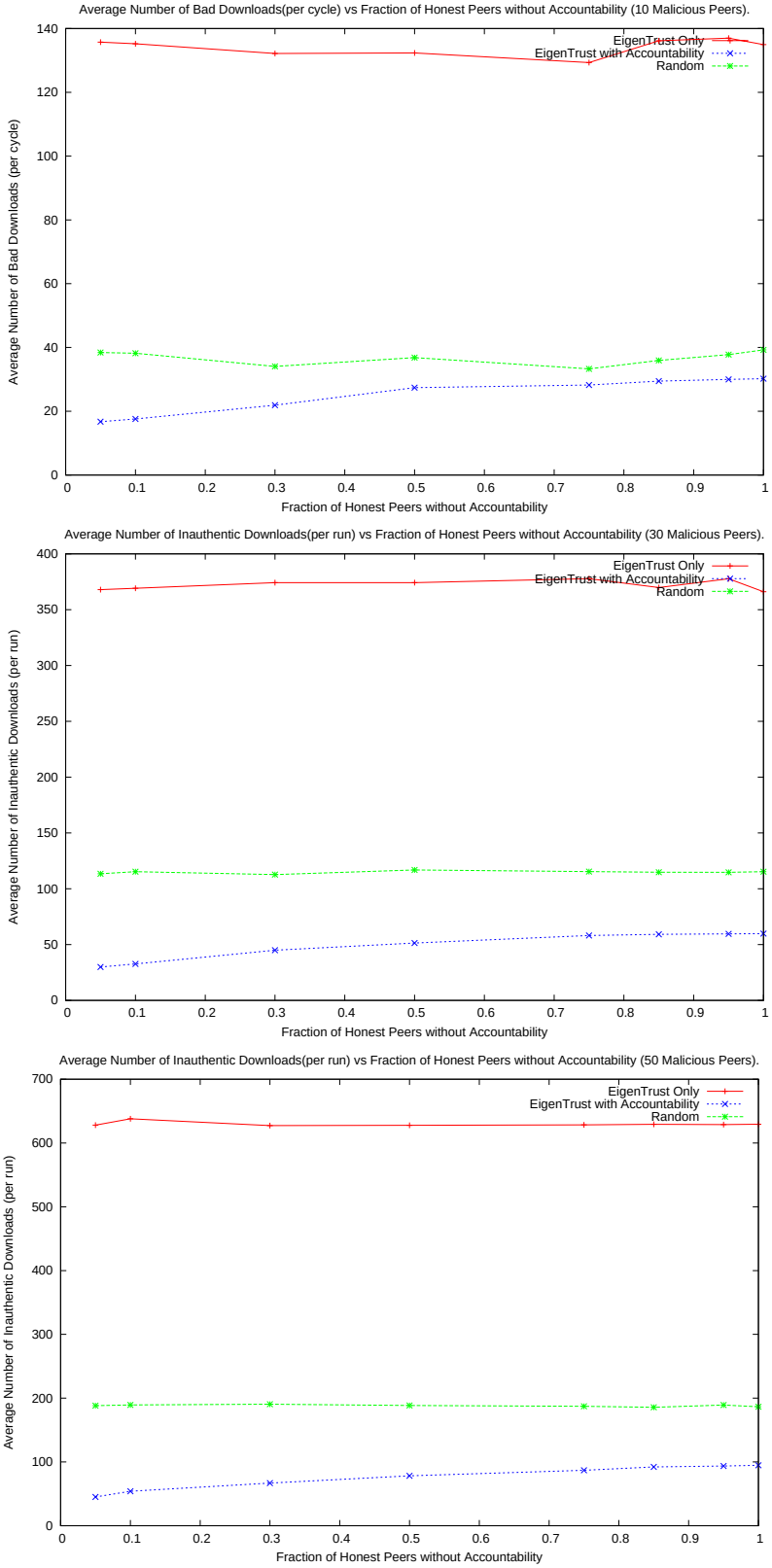

Figure 7.5: Plot showing how β affects effectiveness.

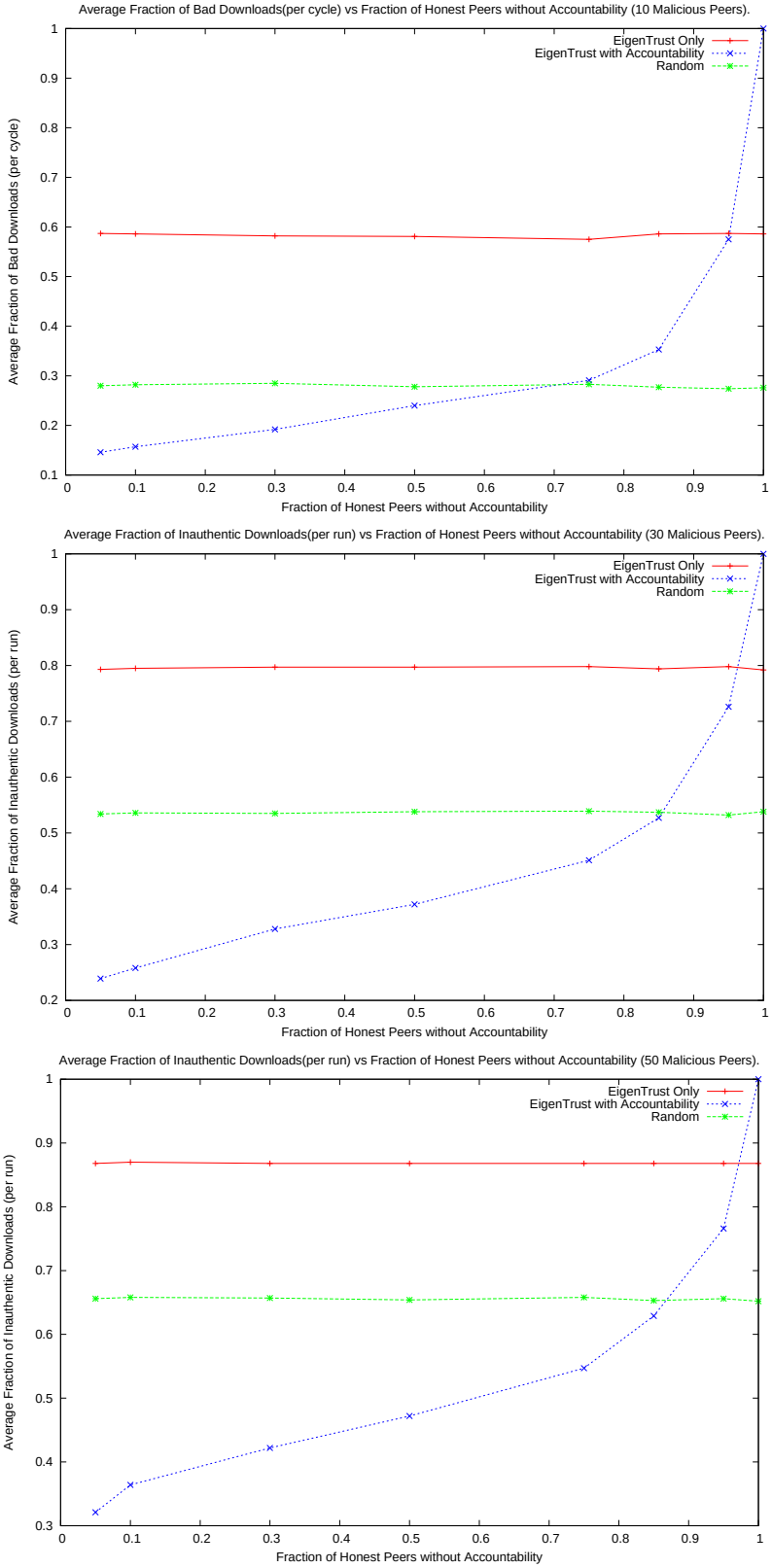

Figure 7.6: Plot showing how β affects robustness.

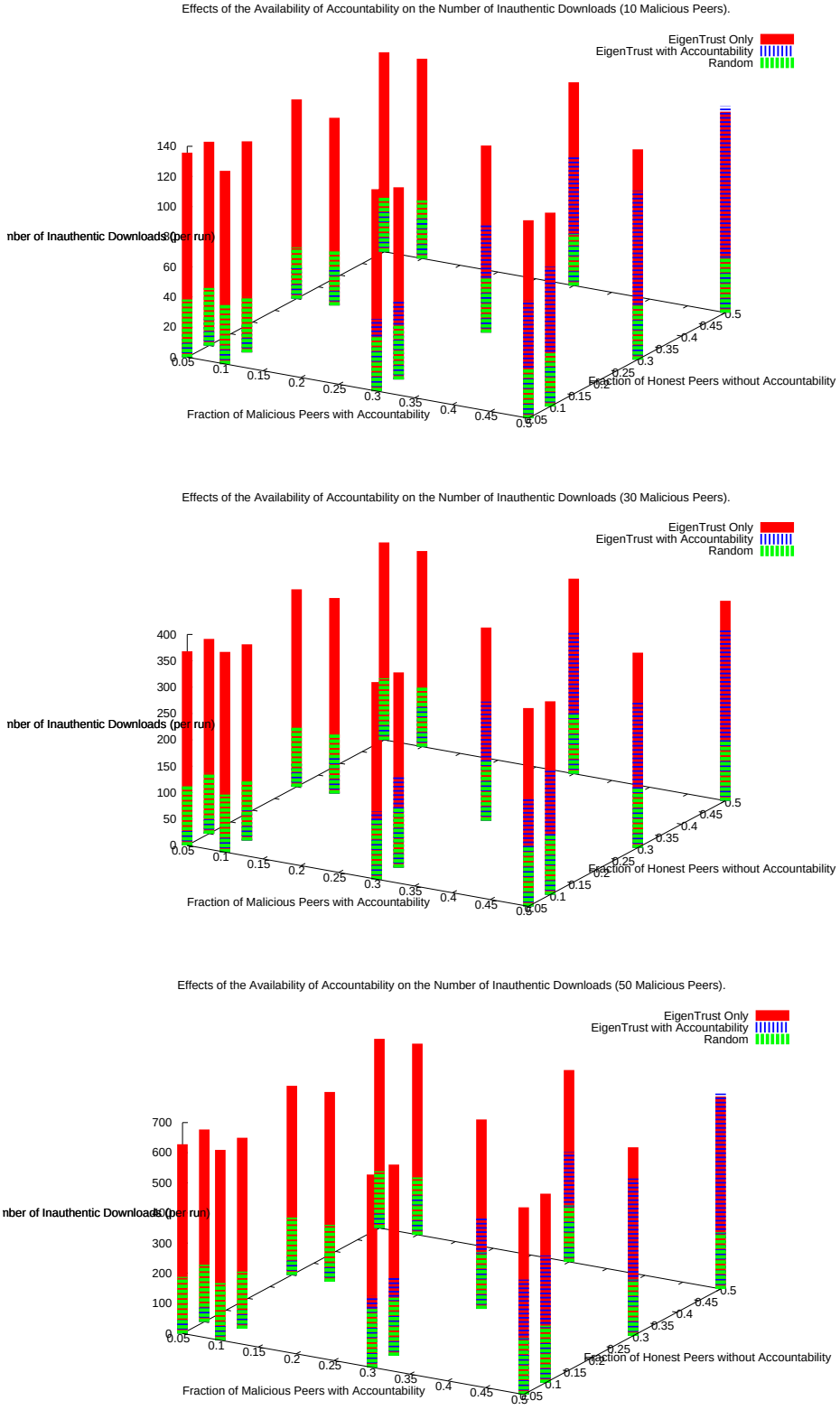

Figure 7.7: Plot showing how the combination of α and β affects effectiveness.

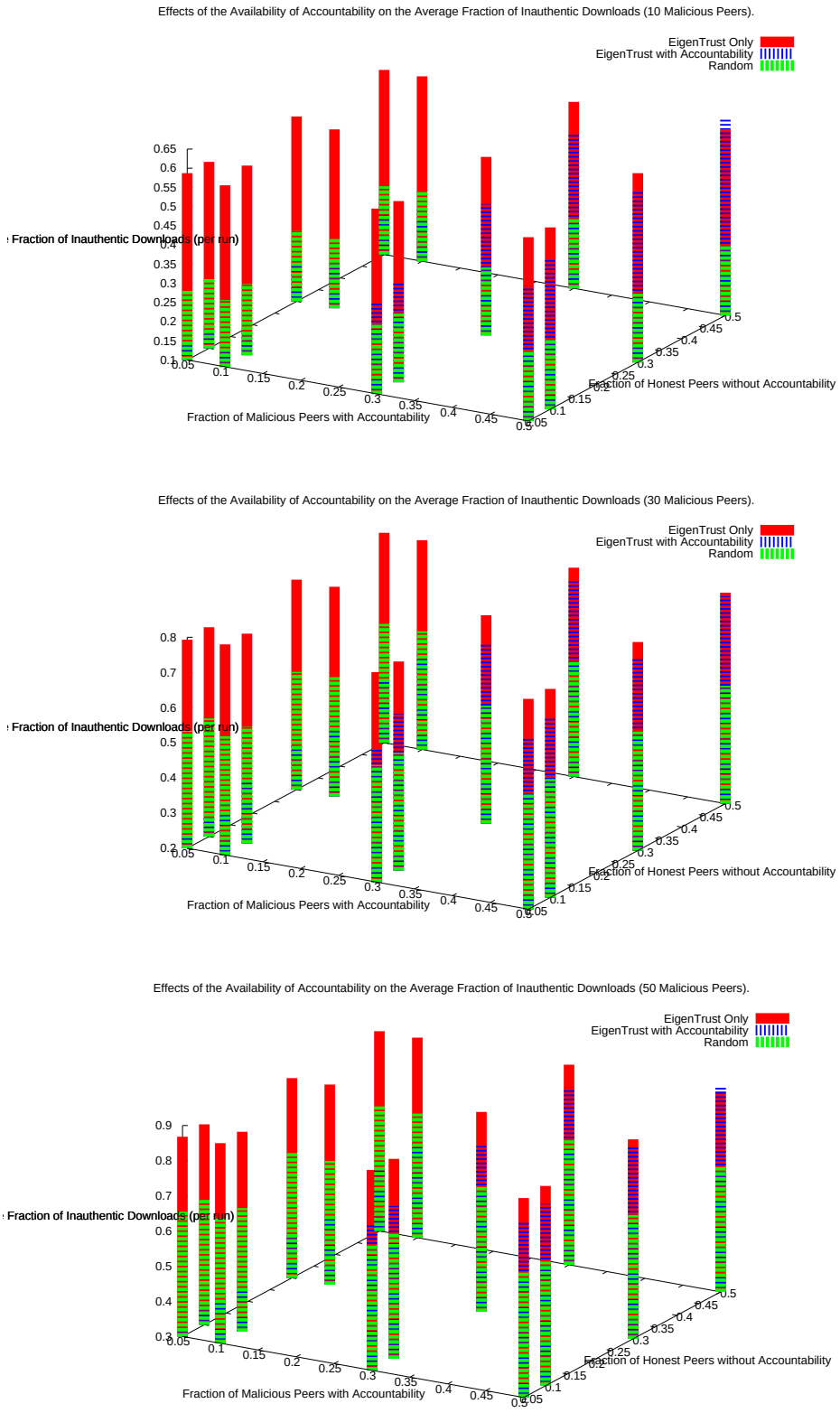

Figure 7.8: Plot showing how the combination of α and β affects robustness.

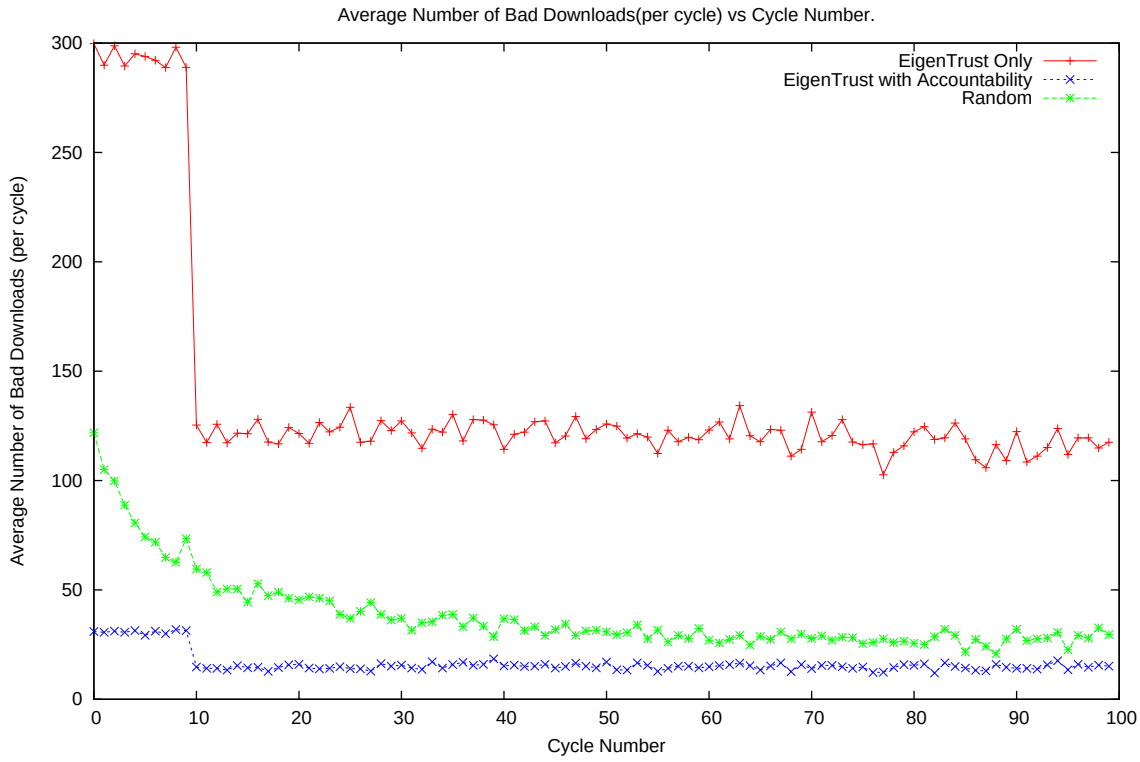

Figure 7.9: Plot showing the effects of delaying the update of reputation.

Chapter 8

Future Work

This dissertation demonstrates the design of robust trust establishment through the use of accountability. While the simulation results are promising, the effects of using accountability in real world applications are unknown. The next step in this research is to investigate the use of accountability for establishing trust in real world situations.

While the establishment of accountability is facilitated through the use of semantic web enabled web resources, semantic web research is still a relatively new field with numerous ongoing research. The main obstacle of using semantic web techniques today is the relatively low adoption rate outside of the research community. Besides using semantic web techniques, other technologies may be explored to seek additional techniques for establishing accountability.

In this dissertation, the main problem of using reputation is that it is easy for adversaries to build reputations. Future research efforts may investigate methods to represent reputation as a computational concept, such that it is more difficult for adversaries to build good reputations. For example, current reputation provides a single dimensional measure of behavior. This allows adversaries to build good reputations by actually behaving well initially. However, in real life, reputations

may not be easily built. For example, reputations of professional workers are built by performing quality work making it difficult for someone to gain reputation by doing a relatively small amount of work. One way to mitigate this problem is to view reputation in multiple aspects. For example, it has been previously mentioned that a FOAF profile can be extended to include the PGP keys of the profile owner. This effectively allows one to tap into the PGP web of trust to obtain the trustworthiness of the profile owner within the PGP web of trust. Similarly, the FOAF profile may also be linked to some professional network, such as LinkedIn. In this situation, when presented with the URI of a peer's FOAF profile, one may access information in other social networks to evaluate the reputation of the peer. This attempts to build a global profile of a principal. By doing so, a bad reputation in one network can affect a principal's reputation in other networks. This increases the social cost of committing malicious acts.

Another possible future research direction is to study ways to improve the performances of trust establishment. One idea is to look into ways to participants in decentralized networks to self-regulate and to self-organize. For example, one may utilize the distance between two peers in social networks as an estimate of trustworthiness. Two peers trust each other if they are neighbors in a social network, otherwise, the value of trust decays as the distance increases. Using this measure, one can select its neighbors in decentralized networks based on these trust estimates. Consider the following scenario where peer p chooses to connect to q, who in turn, chooses to connect to r. Suppose p issues a query to download a file, the query will reach r through q. Suppose r responses to the query and sends the response to p

through q. Since q is connected to r, q considers r to be trustworthy and when q forwards r's response to p, q is effectively vouching for r (if q does not trust r, then q will not be connected to r and will not be able to forward r's response). Similarly, since p is connected to q, p trusts the response forwarded by q. There are two advantages to this form of self-organization. First, malicious peers may be isolated since malicious peers are surrounded by friends. Second, it may be easier to establish accountability since peers are surrounded by other peers who are related through some social relationships. These social relationships can be used for establishing the physical identities of the online peers.

Chapter 9

Conclusion

Trust establishment in decentralized networks is a challenging task due to their open nature where anyone can participate freely. It is difficult to enforce access control to keep out malicious users. Moreover, the problem is compounded by the lack of centralized authority to enforce certain rules and regulations, including managing digital identities and credentials. While the common solution is to turn to reputation systems for trust establishment, this dissertation has shown that relying on reputation alone is not a secure solution. This work contributes to the advancement of trust establishment for decentralized networks in two areas: reducing the complexity of designing trust establishment and improving the security of trust establishment.

Trust establishment is a complex task with many factors, such as security, credentials, and policies to consider. An important contribution of this dissertation is to provide a simple abstract model for trust establishment, as defined by a layered model in Chapter 4. The model abstracts the complex task of trust establishment into simpler components. The abstraction of trust establishment is based on the role of information in trust establishment. The primitive layer addresses the collection of information, while the authentication and provenance layer verifies authenticity and

accuracy of the information respectively. The semantic layer interprets the meaning of the information, such as interpreting the meaning of reputation scores. Finally, the decision layer determines whether trust is established or not. With these layers of abstraction, the model identifies the important issues that should be addressed when designing trust establishment solutions.

The model also recognizes that trust establishment may be dependent on multiple factors. This is illustrated in Fig. 4.5 where the semantic layer is further decomposed into different trust components. Examples of trust components include reputation, recommendation, competency and accountability. An aggregator is responsible for combining the outputs of these trust components. An advantage of this model is that it allows existing trust establishment methods to be combined easily since each existing method can continue to operate independently. This concept is further demonstrated in this dissertation, where two different trust components, reputation and accountability, are combined to achieve a more robust solution for establishing trust in decentralized networks.

Security itself is a complex topic. The meaning of security differs according to contexts. To be able to discuss security for trust establishment in a meaningful way, it is necessary to define what security means in the context of trust establishment. Unfortunately, much existing research defines security based on specific attacks. While these definitions are valid, the trust establishment model is a generic model, and therefore, it is necessary to define security in a generic manner. The trust establishment model contributes to understanding security by identifying the areas in which security has to be addressed, and the type of security that is needed.

In the case of the trust establishment model, the authentication layer ensures that information is originated from a specific source and has not been modified by any other entities. However, authentication does not guarantee that the content of a piece of information is correct. This is an important issue for open, decentralized networks, since the participants may generate false information to gain advantages. Since the semantic layer is dependent on the accuracy of the information, the security requirement at the semantic layer can be identified by how adversaries can manipulate information to gain trust. Therefore, this dissertation focuses on security at the semantic layer. This dissertation contributes to the security of the semantic layer in three aspects. First, the meaning of security is defined. Second, analytical techniques are presented to discover security threats. Third, this dissertation demonstrates the design of a secure trust establishment method for a generic form of file sharing application in decentralized P2P networks.

Robustness is used as a metric for measuring the ability of a trust establishment method to distinguish an honest principal from a malicious principal. A trust establishment is robust if it can distinguish an honest principal from a malicious principal with a high enough probability. From the point of view of an adversary, a robust method requires more effort to gain trust. The measure of robustness thus serves as the definition of security for trust establishment. As a result of this contribution, it is possible to have a quantifiable measurement of security that can be used for evaluating and comparing different trust establishment methods. This is demonstrated in Chapter 5 where it is shown that the use of reputation alone is not robust.

Having defined security for trust establishment in terms of robustness, this dissertation proceeds to describe techniques for measuring the robustness of trust establishment methods. ExBAN serves as an analytical tool for analyzing the credibility of the information used at the semantic layer. ExBAN is an extension of BAN, which is based on belief logic. While BAN has been widely used for analyzing authentication protocols to ensure the authenticity and integrity of information, BAN does not deal with the semantic correctness of the information. The contribution of ExBAN is to fill this void. As a proof of concept, ExBAN is used to demonstrate that reputation alone is not robust in Chapter 5.

Based on the analytical results from ExBAN, an adversarial behavior model can be constructed. This model describes the capabilities of the adversaries with respect to a trust establishment method. A feature of this model is the use of generic classes of attacks to describe the capabilities of the adversaries. This is an important feature because there are too many types of attacks that are available to the adversaries. Therefore, it is impractical to describe the capabilities of an adversary based on each and everyone of them. Moreover, there are possible attacks in the future that are now unknown. Attacks are classified under different classes according to their effects. Attacks belonging to the same class have the same effects. For example, this dissertation shows that an adversary model for a reputation system consists of: manipulating reputation scores, failure to deliver services and to delay updates of reputation. An additional advantage of using attack classes to describe an adversary is that when a new type of attack surfaces in the future, its impact to a trust establishment method is already known if the new attack belongs to one of

the attack classes described in the adversarial behavior model.

Designing a robust trust establishment method can be a difficult process. The adversary model contributes to the design of robust trust establishment methods in two ways. First, by describing the capabilities of an adversary, the pitfalls are known in advance and additional methods can be deployed to address these pitfalls. Second, and more important, it provides insights into ways of improving the robustness of trust establishment methods by making use of actions that are beyond the capabilities of the adversaries. The use of accountability to complement reputation systems is an example of such design strategy.

While an adversary can gain trust by accumulating sufficient reputation, the lack of centralized authorities in decentralized networks means that even when adversaries are discovered, there are no authorities to prosecute them. Accountability addresses these issues. First, even though decentralized networks represent open world models, the users belong to lawful societies and have to answer to their online behaviors. By establishing a link between an online identity to its corresponding physical identity, accountability can be enforced. Moreover, it is difficult for adversaries to establish accountability since they do not want to be held responsible for their malicious activities. Therefore, a trust establishment method can make use of the establishment of accountability to distinguish between honest and malicious principals. To establish accountability, this dissertation proposes the use of social profiles on the Internet.

The effects of using accountability to reinforce reputations are studied in a series of simulations. Unlike many other works where simulation scenarios are cho-

sen in an ad hoc manner, the simulation scenarios in this research are chosen based on the adversary model from ExBAN analysis. The adversary model describes the capabilities of the adversaries in attacking reputation systems in P2P file sharing applications. These capabilities are reflected in the design of the simulation scenarios. As described in Chapter 7, simulation results show that in the presence of adversaries, the combination of reputation and accountability is more robust than the use of reputation alone.

With computing trends moving towards an open, ubiquitous and pervasive environment, trust establishment becomes critical to secure computing. The results presented in this dissertation provide a promising step towards achieving the goal of improving the security of computers in such environments by providing techniques that contribute to the design of robust trust establishment methods.

Bibliography

[1] IEEE Std 610.12-1990 - IEEE standard glossary of software engineering terminology, 1990.

[2] TCG TPM specification version 1.2, 2003. `http://www.trustedcomputing.org`.

[3] A. Abdul-Rahman and S. Hailes. A distributed trust model. In *Proceedings of the Workshop on New Security Paradigms*, pages 48–60, 1997.

[4] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. In *33rd Hawaii International Conference on System Sciences*, 2000.

[5] K. Aberer, P. Cudré-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva, and R. Schmidt. P-Grid: A self-organizing structured p2p system. *SIGMOD Record*, 32(2), September 2003.

[6] K. Aberer and Z. Despotovic. Managing trust in a peer-2-peer information system. In *Tenth International Conference on Information and Knowledge Management (CIKM01)*, pages 310–317. ACM Press, 2001.

[7] P. Bajari and A. Hortacsu. Winner's curse, reserve prices and endogenous entry: Empirical insights from ebay auctions. *RAND Journal of Economics*, 34(2), 2003.

[8] B. Barber. *Logic and Limits of Trust*. Rutgers University Press, 1983.

[9] T. Beth, M. Borcherding, and B. Klein. Valuation of trust in open networks. In *3rd European Symposium on Research in Computer Security - ESORICS'94*, pages 3–18, 1994.

[10] M. Blaze. A cryptographic file system for UNIX. In *Proceedings of the ACM Conference on Computer and Communications Security*, pages 9–16, 1993.

[11] M. Blaze, J. Feigenbaum, and A. Keromytis. Keynote: Trust management for public-key infrastructures. In *Security Protocols International Workshop*, pages 59–63, Cambridge, England, Apr 1998. Springer LNCS.

[12] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized trust management. In *IEEE Symposium on Security and Privacy*, Oakland, USA, 1996.

[13] D. Brickley and L. Miller. *FOAF Vocabulary Specification 0.91*, 2007. `http://xmlns.com/foaf/spec`.

[14] M. Burrows, M. Abadi, and R. Needham. The logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, 1990.

[15] J. Carroll, C. Bizer, P. Hayes, and P. Stickler. Named graphs, provenance and trust. In *Proceedings of the 14th International Conference on World Wide Web*, pages 613–622, Chiba, Japan, 2005.

[16] B. Christianson and W. Harbison. Why isn't trust transitive? In *Security Protocols International Workshop*, 1996.

[17] Y.H. Chu, J. Feigenbaum, B. LaMacchia, P. Resnick, and M. Strauss. REFEREE: Trust management for web applications. *The World Wide Web Journal*, 2(3):127–139, 1997.

[18] B. Cohen. Incentives build robustness in BitTorrent. Technical report, 2003.

[19] F. Cornelli, E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati. Choosing reputable servents in a p2p network. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 376–386, Honolulu, Hawaii, USA, 2002.

[20] M. Deutsch. Cooperation and trust. In M.R. Jones, editor, *Nebraska Symposium on Motivation*. Nebraska University Press, 1962.

[21] C. Dong, G. Russello, and N. Dulay. Trust transfer in distributed systems. In *Proceedings of 2007 Joint iTrust and PST Conferences on Privacy, Trust Management and Security (IFIPTM 2007)*, Moncton, Canada, 2007.

[22] J. Douceur. The Sybil Attack. In *IPTPS02 Workshop, Cambridge*, MA (USA), Mar 2002.

[23] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. *SPKI Certificate Theory*. RFC 2693, IETF, September 1999.

[24] A. Fernandes, E. Kotsovinos, S. Ostring, and B. Dragovic. Pinocchio: Incentives for honest participation in distributed trust management. In *Proceedings of the 2nd International Conference on Trust Management (iTrust 2004)*. Spriger-Verlag LNCS, 2004.

[25] E. Friedman and P. Resnick. The social cost of cheap pseudonyms. *Journal of Economics and Management Strategy*, 10(2):173–199, 2001.

[26] D. Gambetta. *Trust*. Blackwell, 1990.

[27] Y. Gil and V. Ratnakar. Trusting information sources one citizen at a time. In *First International Semantic Web Conference*, Sardinia, Italy, June 2002.

[28] J. Golbeck and J. Hendler. Inferring trust relationships in web-based social networks. *ACM Transactions on Internet Technology*, 7(1), 2005.

[29] J. Golbeck and J. Hendler. FilmTrust: movie recommendations using trust in web-based social networks. In *IEEE Consumer Communications and Networking Conference*, Jan 2006.

[30] L. Gong, R. Needham, and R. Yahalom. Reasoning about belief in cryptographic protocols. In *IEEE Symposium on Research in Security and Privacy*, pages 234–248, Oakland, California, May 1990.

[31] T. Grandison and M. Sloman. A survey of trust in internet application. *IEEE Communications Surveys, Fourth Quarter*, 2000.

[32] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proceedings of the 13th International Conference on World Wide Web*, pages 403–412, New York, NY, USA, 2004.

[33] A. Herzberg, Y. Mass, J. Michaeli, Y. Ravid, and D. Naor. Access control meets public key infrastructure, or: Assigning roles to strangers. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, 2000.

[34] C.A.R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–585, October 1969.

[35] D. Houser and J. Wooders. Reputation in auctions: Theory, and evidence from ebay. *Journal of Economics and Management Strategy*, 15:353–369, 2006.

[36] R. Housley, W. Polk, W. Ford, and D. Solo. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile.* RFC 3280, IETF, April 2002.

[37] IBM. IBM trust establishment policy language. `http://www.haifa.ibm.com/projects/software/e-Business/TrustManager/PolicyLanguage.html`.

[38] A.J.I. Jones and M.J. Sergot. A formal characterisation of institutionalised power. *Journal of the Interest Group in Pure and Applied Logics*, 4(3):429–445, June 1996.

[39] A. Jøsang. The right type of trust for distributed system. In *Workshop on New Security Paradigms*, pages 119–131, Lake Arrowhead, CA, USA, 1996.

[40] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Twelfth International World Wide Web Conference*, 2003.

[41] G. Kan. Gnutella. In A. Oram, editor, *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly Media, Inc, 2001.

[42] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[43] A. Langley. Freenet. In A. Oram, editor, *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly Media, Inc, 2001.

[44] R. Levien and A. Aiken. Attack-resistant trust metrics for public key certification. In *Proceedings of the 7th Conference on USENIX Security Symposium*, pages 229–242, San Antonio, Texas, 1998. USENIX Association.

[45] D. Lucking-Reiley, D. Bryan, N. Prasad, and D. Reeves. Pennies from ebay : The determinants of price in online auctions. *Journal of Industrial Economics*, 55(2):223–233, 2007.

[46] N. Luhmann. Trust: A mechanism for the reduction of social complexity. In *Trust and Power: Two works by Niklas Luhmann*. John Wiley and Sons, 1979.

[47] S. Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, Department of Mathematics and Computer Science, University of Stirling, 1994.

[48] P. Massa and B. Bhattacharjee. Using trust in recommender systems: An experimental analysis. In *Proceedings of the 2nd International Conference on Trust Management*. Springer–Verlag LNCS, 2004.

[49] R. Matthew, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *Proceedings of the Second International Semantic Web Conference*, Sanibel Island, Florida, USA, 2003.

[50] A. Medina, I. Matta, and J. Byers. On the origin of power laws in internet topologies. *Technical report*, 2000.

[51] R. Needham and M. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978.

[52] B. Neuman and T. Ts'o. Kerberos: An authentication service for computer networks. *IEEE Communications*, 32(9):33–38, September 1994.

[53] M. O'Keefe. A universal access smart-card-based secure file system. In *Proceedings of the 3rd Annual Conference on Atlanta Linux Showcase*, Atlanta, Georgia, USA, 1999. USENIX Association.

[54] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: bringing order to the web. *Technical report*, 1998.

[55] E. Pavlov, J. Rosenschein, and Z. Topol. Supporting privacy in decentralized additive reputation systems. In *Proceedings of the Second International Conference on Trust Management (iTrust 2004)*, pages 108–119. Springer-Verlag LNCS, 2004.

[56] N. L. Petroni, T. Fraser, J. Molina, and W. A. Arbaugh. Copilot - a coprocessor-based kernel runtime integrity monitor. In *USENIX Security Symposium*, pages 179–194, Aug 2004.

[57] P. Resnick and H. Varian. Recommender systems. *Communications of the ACM*, 40(3), March 1997.

[58] P. Resnick and R. Zeckhauser. Trust among strangers in internet transactions: Empirical analysis of eBay's reputation system. In Michael R. Baye, editor, *The Economics of the Internet and E-Commerce*, volume 11 of *Advances in Applied Microeconomics*, Amsterdam, 2002. Elsevier Science.

[59] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation systems. *Communications of the ACM*, 43(12):45–48, December 2000.

[60] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[61] R. Sailer, X. Zhang, T. Jaeger, and L. van Doorn. Design and implementation of a tcg-based integrity measurement architecture. In *USENIX Security Symposium*, pages 223–238, Aug 2004.

[62] S. Saroiu, P. Gummadi, and S. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networking (MMCN '02)*, San Jose, CA, USA, 2002.

[63] M. Schlosser, T. Condie, and S. Kamvar. Simulating a p2p file-sharing network. In *Proceedings of the First Workshop on Semantics in P2P and Grid Computing*, 2002.

[64] J. Seigneur, A. Gray, and C. Jensen. Trust transfer: Encouraging self-recommendations without sybil attack. In *Proceedings of the Third International Conference on Trust Management (iTrust 2005)*, pages 321–337, Paris, France, 2005. Springer-Verlag LNCS.

[65] J. Seigneur and C. Jensen. Trading privacy for trust. In *Proceedings of the 2nd International Conference on Trust Management (iTrust 2004)*. Springer-Verlag LNCS, 2004.

[66] E. Shi, A. Perrig, and L. van Doorn. Bind: A fine-grained attestation service for secure distributed systems. In *IEEE Symposium on Security and Privacy*, Oakland, CA, May 2005.

[67] C. Shirky. Listening to Napster. In A. Oram, editor, *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly Media, Inc, 2001.

[68] A. Singh and L. Liu. Trustme: Anonymous management of trust relationships in decentralized p2p systems. In *P2P '03: Proceedings of the 3rd International Conference on Peer-to-Peer Computing*, 2003.

[69] M. K. Smith, C. Welty, and D. L. McGuinness. *OWL Web Ontology Language Guide*. World Wide Web Consortium, Feb 2004. `http://www.w3.org/TR/owl-guide/`.

[70] P. Syverson and P. van Oorschot. On unifying some cryptographic protocol logics. In *IEEE Computer Society Symposium on Research in Security and Privacy*, pages 14–28, Oakland, California, May 1994.

[71] D. Martin *et. al.* *OWL-S: Semantic Markup for Web Services*. World Wide Web Consortium, 2004. `http://www.w3.org/Submission/OWL-S`.

[72] H. Tran, M. Hitchens, V. Varadharajan, and P. Watters. A trust based access control framework for p2p file-sharing systems. In *HICSS '05: Proceedings of the Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, 2005.

[73] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[74] D. Weitzner, J. Hendler, T. Berners-Lee, and D. Connolly. Creating a policy-aware web: Discretionary, rule-based access for the world wide web. In E. Ferrari and B. Thuraisingham, editors, *Web and Information Security*. IRM Press, 2006.

[75] W. Winsborough and N. Li. Towards practical automated trust negotiation. In *Third International Workshop on Policies for Distributed Systems and Networks*, pages 92–103. IEEE Computer Society Press, 2002.

[76] W. Winsborough, K. Seamons, and V. Jones. Automated trust negotiation. In *DARPA Information Survivability Conference and Exposition*, pages 88–102. IEEE Press, 2000.

[77] M. Winslett. An introduction to trust negotiation. In *iTrust 2003, LNCS*, volume 2692, pages 275–283. Springer, 2003.

[78] L. Xiong and L. Liu. A reputation-based trust model for peer-to-peer ecommerce communities. In *IEEE Conference on ECommerce (CEC'03)*, 2003.

[79] R. Yahalom, B. Klein, and T. Beth. Trust relationships in secure systems - a distributed authentication perspective. In *IEEE Symposium on Security and Privacy*, 1993.

[80] Z. Yu, C.Y. Seng, T. Jiang, X. Wu, and W. Arbaugh. Robust routing in malicious environment for ad hoc networks. In *First Information Security Practice and Experience Conference, Lecture Notes in Computer Science 3439*, pages 36–47, Singapore, April 2005. Springer-Verlag.

[81] C. Ziegler and J. Golbeck. Investigating correlations of trust and interest similarity – do birds of a feather really flock together? *Decision Support Systems*, 43(2), 2006.

[82] C. Ziegler and G. Lausen. Analyzing correlation between trust and user similarity in online communities. In *Proceedings of the 2nd International Conference on Trust Management*, pages 251–265. Springer-Verlag LNCS, 2004.

[83] P. Zimmermann. *The Official PGP User's Guide*. MIT Press, May 1995.